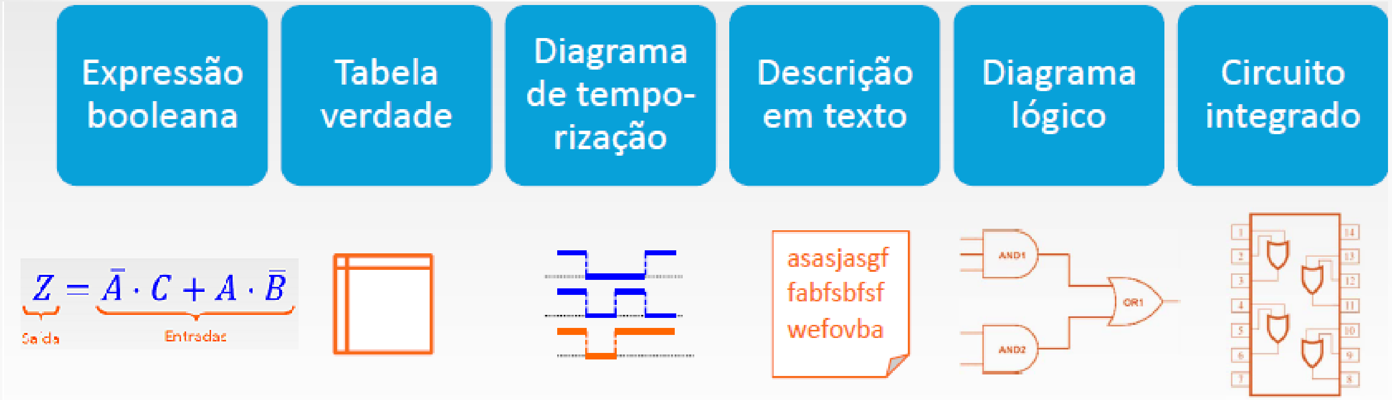
Circuitos Digitais

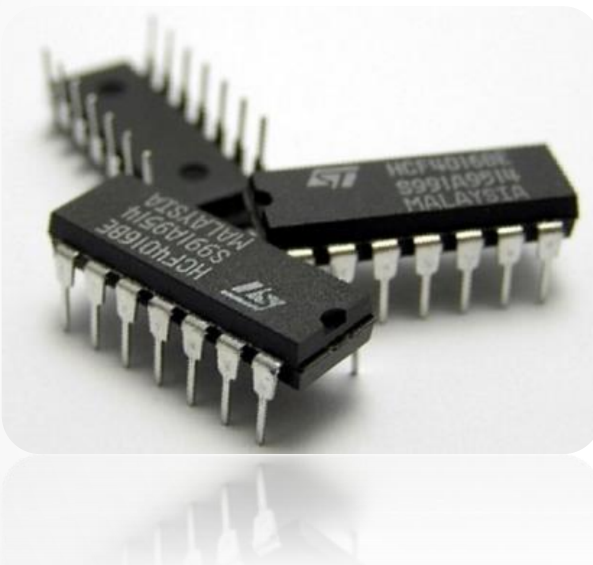
Engenharia Elétrica/Engenharia de Automação/
Engenharia de Computação/Sistemas de Informação/
Ciência da Computação/Tecnologia de Redes

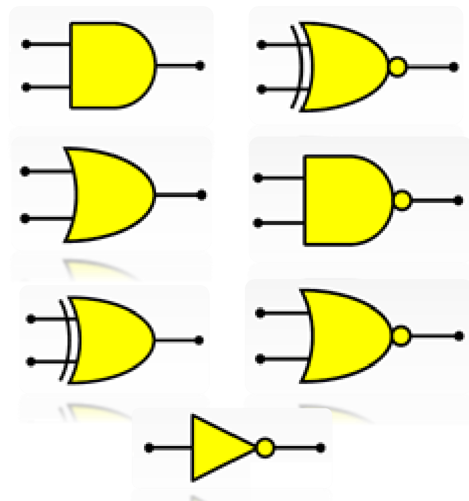
FAESA

Prof. VICTOR MARQUES MIRANDA

CONTEÚDOS







Unidade 3

Portas Lógicas e Formas de Representação Lógica

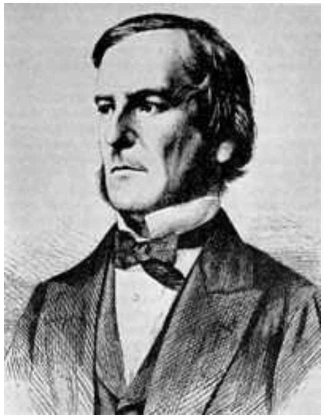
Objetivos

- ✓ Portas Lógicas
  - ✓ Tipos de Portas Lógicas e a Lógica Booleana
- ✓ Formas de Representação de uma Função Lógica
  - ✓ Obtenção de Expressões Booleanas a partir de um Circuito Lógico
  - ✓ Obtenção de Circuitos a partir de Expressões Booleanas
  - ✓ Obtenção da Tabela Verdade a partir de uma Expressão
  - ✓ Obtenção da Tabela Verdade a partir de Circuitos Lógicos
  - ✓ Formas Canônicas

Portas Lógicas e a Lógica Booleana

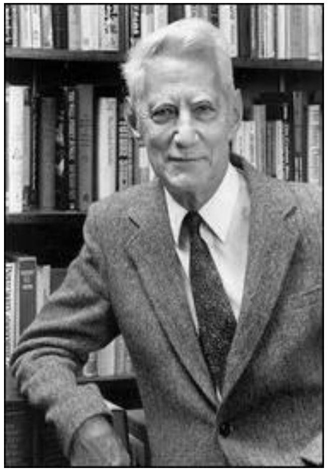
Histórico

- Em meados do século XIX o matemático inglês George Boole desenvolveu um sistema matemático de análise lógica, conhecido como Álgebra de Boole.



George Boole (1815-1864)

- Em meados do século XX, o americano Claude Elwood Shannon sugeriu que a Lógica Booleana poderia ser usada para análise e projeto de circuitos de comutação.



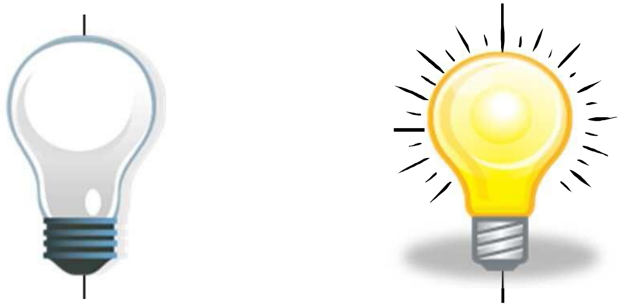
Claude Elwood Shannon (1916-2001)

Histórico

- Nos primórdios da eletrônica, todos os problemas eram solucionados por meio de **sistemas analógicos.**
- Com o avanço da tecnologia, os problemas passaram a ser solucionados pela **eletrônica digital.**
- Na eletrônica digital, os sistemas (computadores, processadores de dados, sistemas de controle, codificadores, decodificadores, etc) empregam um grupo de circuitos lógicos que constituem a base dos seus projetos, conhecidos como **<u>portas lógicas.</u>**
- Com a utilização adequadas dessas portas é possível **implementar todas as expressões geradas pela álgebra de Boole.**

Lógica Booleana

- Na lógica de Boole, há somente dois estados (valores ou símbolos) permitidos:
  - **Estado 0 (zero)**
  - **Estado 1 (um)**



- Em geral:
  - O estado **0 (zero) representa** **não**, **falso**, aparelho desligado, ausência de tensão, chave elétrica desligada, etc.
  - O estado **1 (um) representa** **sim**, **verdadeiro**, aparelho ligado, presença de tensão, chave ligada, etc.

Álgebra Booleana

- Assim, na álgebra booleana, se representarmos por 0 uma situação, a situação contrária é representada por 1.

- Portanto, em qualquer bloco (porta ou função) lógico somente esses dois estados (0 ou 1) são permitidos em suas entradas e saídas.

- Uma **variável booleana também só assume um dos dois estados permitidos (0 ou 1).**

- Na prática, as variáveis booleanas são muitas vezes usadas para representar o **nível de tensão presente em uma conexão ou em terminais de entrada/saída de um circuito,** chamado **nível lógico.**

Álgebra Booleana

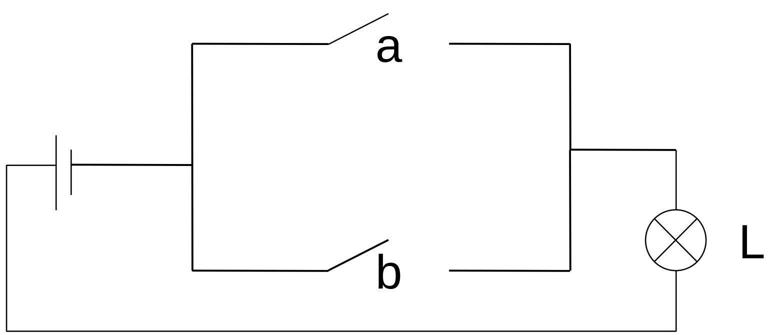
- Por exemplo, em um determinado sistema digital, o **valor booleano 0** pode representar qualquer tensão dentro da faixa de 0V a 0.8V, enquanto o **valor booleano 1** pode representar qualquer tensão dentro da faixa 2 a 5V.

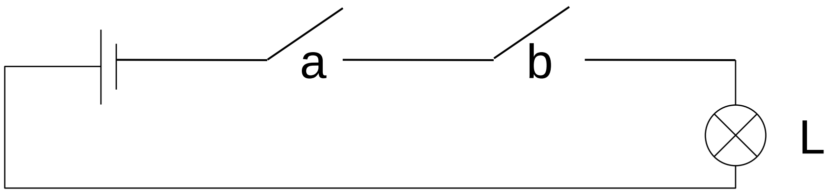
- Logo, usamos frequentemente também as designações:
  - ✓ ***ALTO (HIGH) para nível lógico 1; e***
  - ✓ ***BAIXO (LOW) para o nível 0.***

- Veremos mais detalhes sobre a Álgebra Booleana em outra oportunidade.
- Agora focaremos nosso estudo em torno das **portas e circuitos lógicos**.

Circuitos de Chaveamento

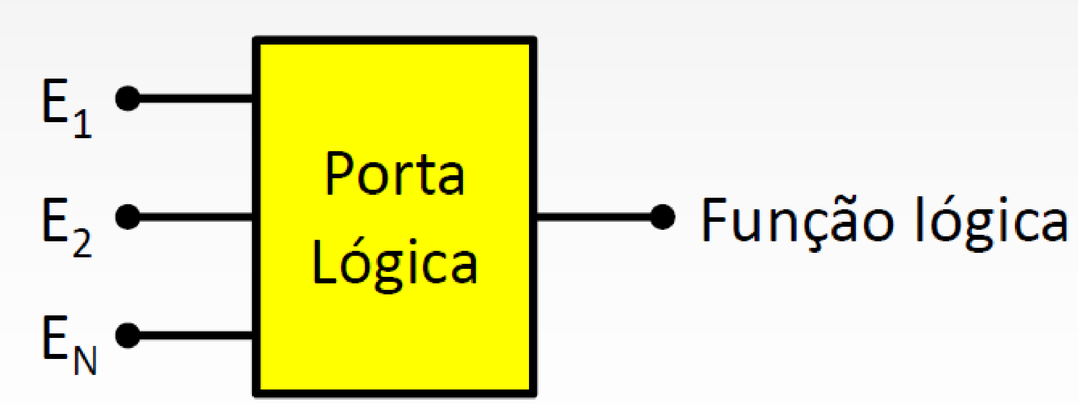
- A Lógica de Boole pode ser aplicada através de **<u>circuitos de chaveamento (ou comutação)</u>**.
- Analise os seguintes circuitos elétricos abaixo considerando a e b como chaves (interruptores) e L (lâmpada).
- Qual a representação da passagem da corrente elétrica nos circuitos abaixo?





Portas Lógicas

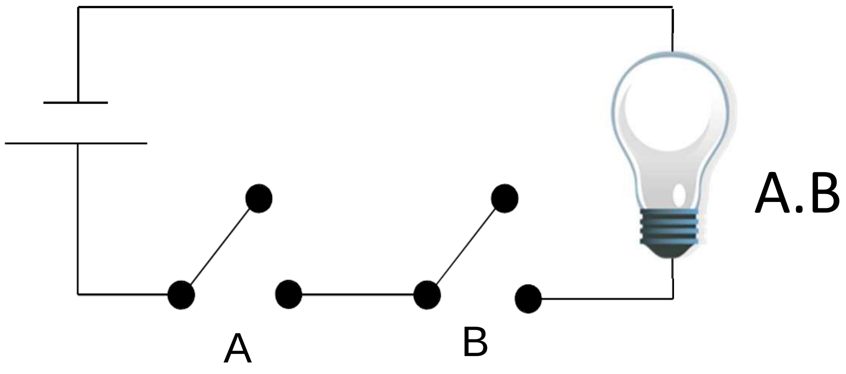
- Uma porta lógica é um circuito digital de dois estados, cuja saída pode ser alta ou baixa, a depender das entradas altas/baixas.
- Literalmente, o termo designa uma porta que só permite a passagem de um sinal elétrico somente se certas condições lógicas são atendidas.



Função E (AND)

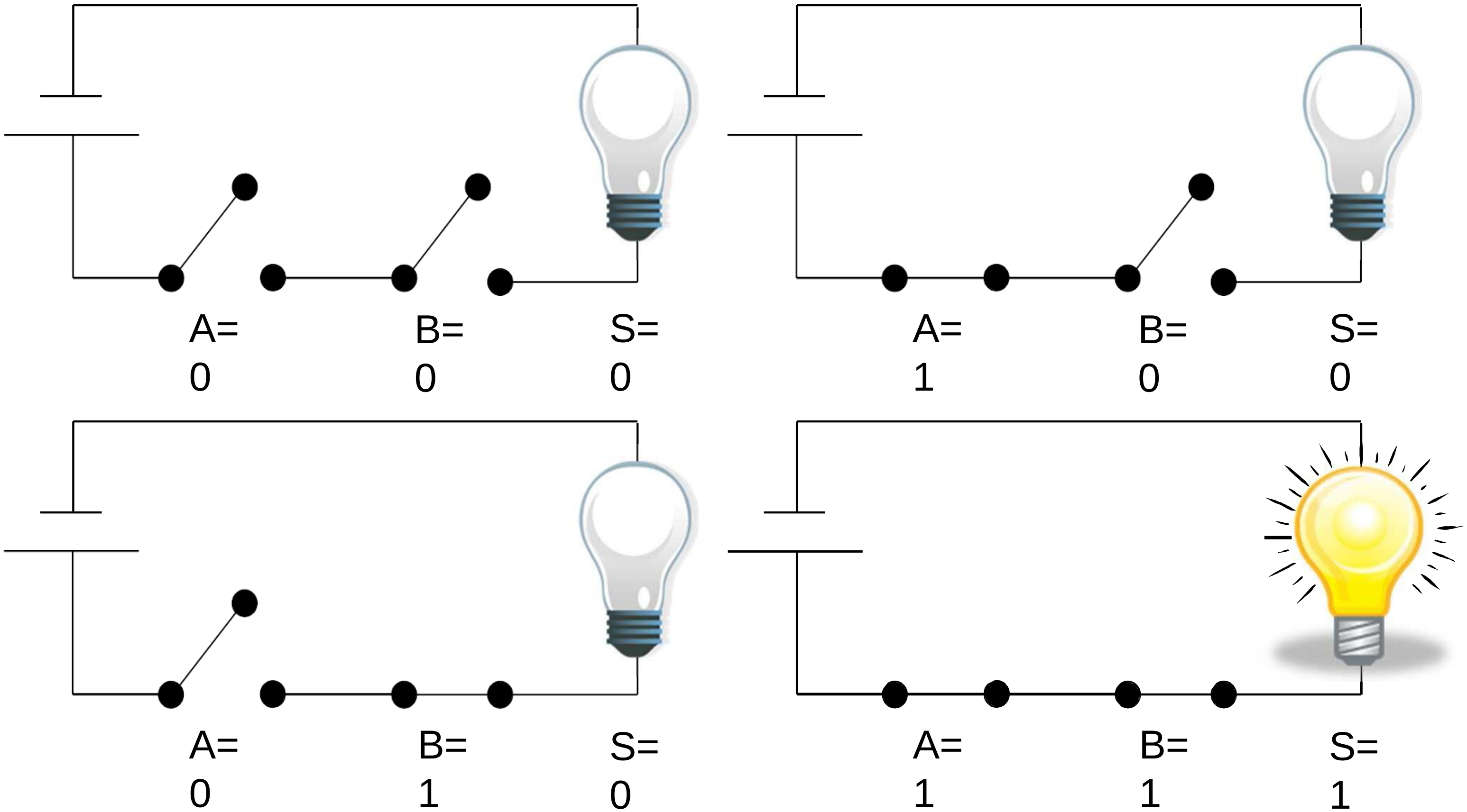
- Executa a **multiplicação (conjunção) booleana** de duas ou mais variáveis binárias.

- Por exemplo, assuma a convenção no circuito:
  - **Chave aberta = 0; Chave fechada = 1**
  - **Lâmpada apagada = 0; Lâmpada acesa = 1**



Função E (AND)

Situações possíveis:



Função E (AND)

- ✓ *Se a chave* ***A está aberta (A=0)*** *e a chave* ***B aberta (B=0),*** *não haverá circulação de energia no circuito, logo* ***a lâmpada fica apagada (S=0).***

- ✓ *Se a chave* ***A está fechada (A=1)*** *e a chave* ***B aberta (B=0),*** *não haverá circulação de energia no circuito, logo* ***a lâmpada fica apagada (S=0).***

- ✓ *Se a chave* ***A está aberta (A=0)*** *e a chave* ***B fechada (B=1),*** *não haverá circulação de energia no circuito, logo* ***a lâmpada fica apagada (S=0).***

- ✓ *Se a chave* ***A está fechada (A=1)*** *e a chave* ***B fechada (B=1)****, haverá circulação de energia no circuito e* ***a lâmpada fica acesa (S=1).***

- ✓ **Conclusão:** observando todas as quatro situações possíveis (interpretações), é possível concluir que **a lâmpada fica acesa somente quando as chaves A e B estiverem simultaneamente fechadas (A=1 e B=1).**

Função E (AND)

Para representar a expressão

- ✓ S = A e B

Adotaremos a representação

- ✓ S = A . B, onde se lê S = A e B

Porém, existem notações alternativas

- ✓ S = A & B
- ✓ S = A , B
- ✓ S = A ∧ B

Tabela Verdade

- A tabela verdade é um mapa onde são colocadas **todas as combinações possíveis para os níveis lógicos presentes nas entradas com o correspondente nível lógico na saída** para uma expressão booleana qualquer.

- Como visto no exemplo anterior, para 2 variáveis booleanas (A e B), há 4 interpretações possíveis.

- Em geral, para ***N* variáveis booleanas de entrada**, há **2^N interpretações possíveis.**

A	B
0	0
0	1
1	0
1	1

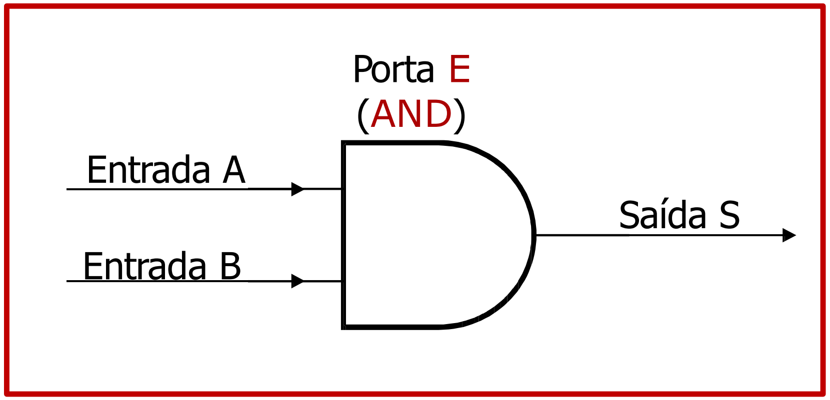
A	B	C
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

Tabela Verdade da Função E (AND)

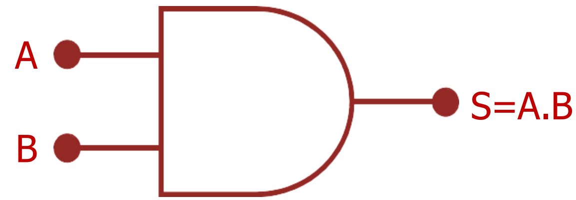
A	B	A .B
0	0	0
0	1	0
1	0	0
1	1	1

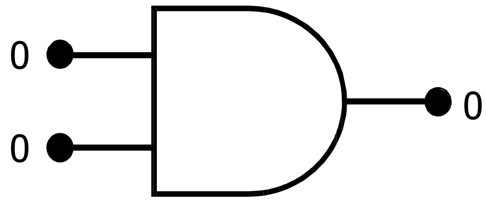
Porta Lógica E (AND)

- A porta E é um circuito que executa a função E
- A porta E executa a tabela verdade da função E
  - Portanto, a saída será 1 somente se ambas as entradas forem iguais a 1; nos demais casos, a saída será 0
- Representação:

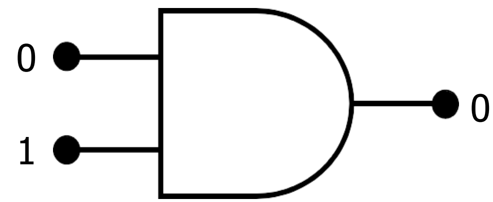


Porta Lógica E (AND)

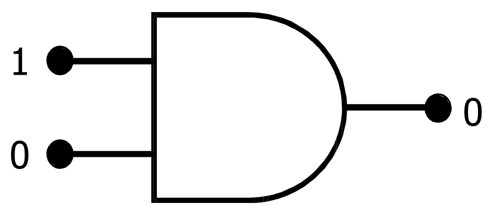




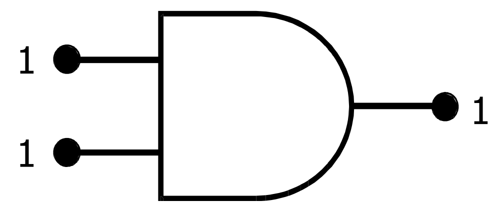
A	B	S=A.B
0	0	0
0	1	0
1	0	0
1	1	1



A	B	S=A.B
0	0	0
0	1	0
1	0	0
1	1	1



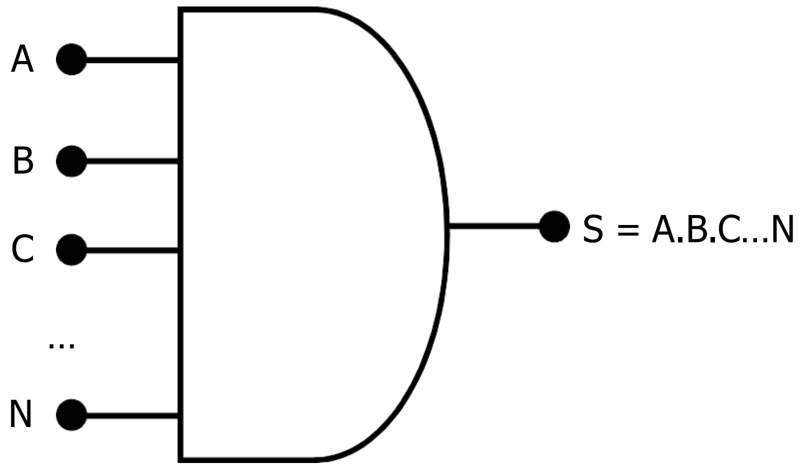
A	B	S=A.B
0	0	0
0	1	0
1	0	0
1	1	1



A	B	S=A.B
0	0	0
0	1	0
1	0	0
1	1	1

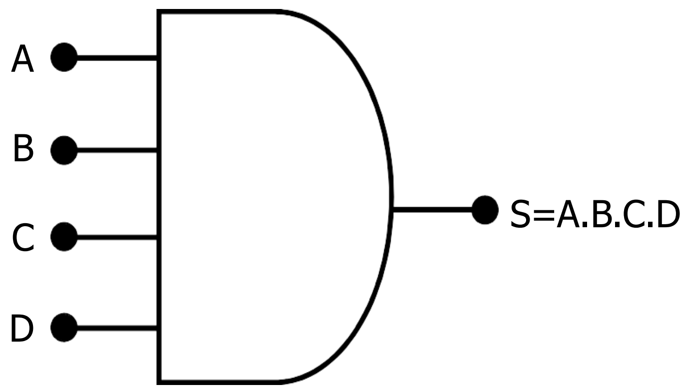
Porta Lógica E (AND)

- É possível estender o conceito de uma porta E para um número qualquer de variáveis de entrada.
- Nesse caso, temos uma porta E com N entradas e somente uma saída.
- A saída será 1 se e somente se as N entradas forem iguais a 1; nos demais casos, a saída será 0.



Porta Lógica E (AND)

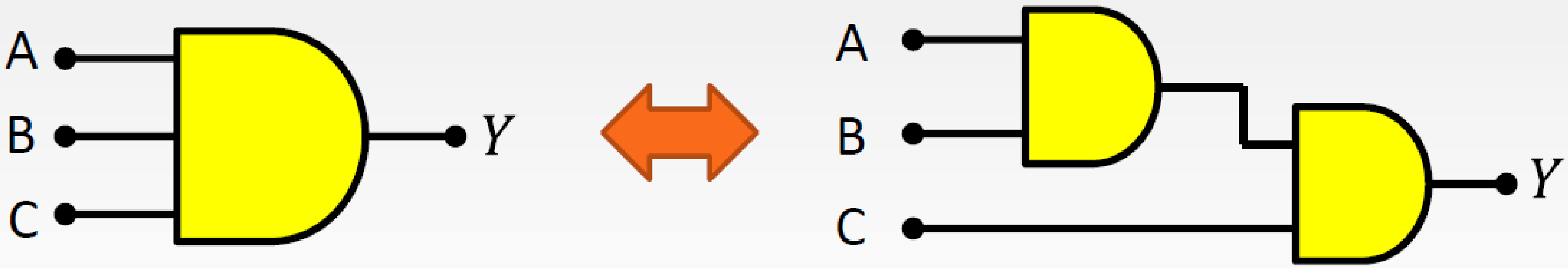
Por exemplo,
$S = A.B.C.D$



A	B	C	D	S
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

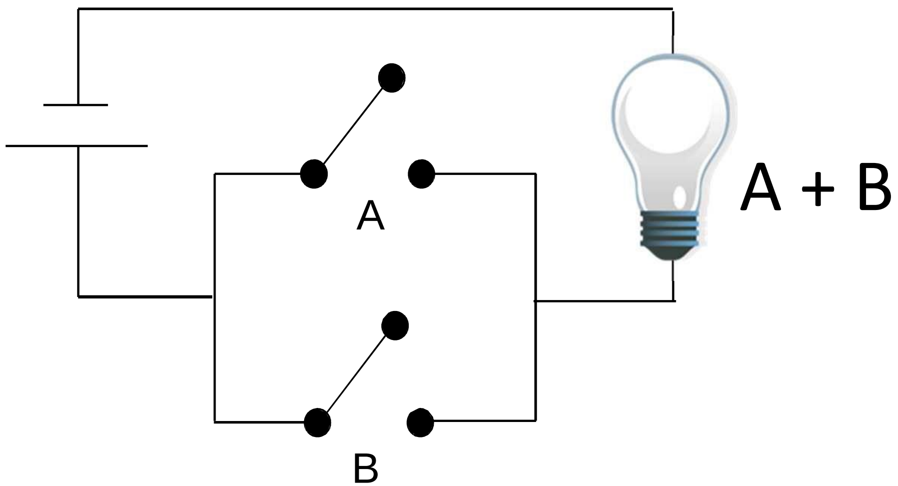
Portas com três ou mais entradas

- Como gerar o mesmo resultado de uma porta AND de três entradas se há disponível apenas portas AND de duas entradas?

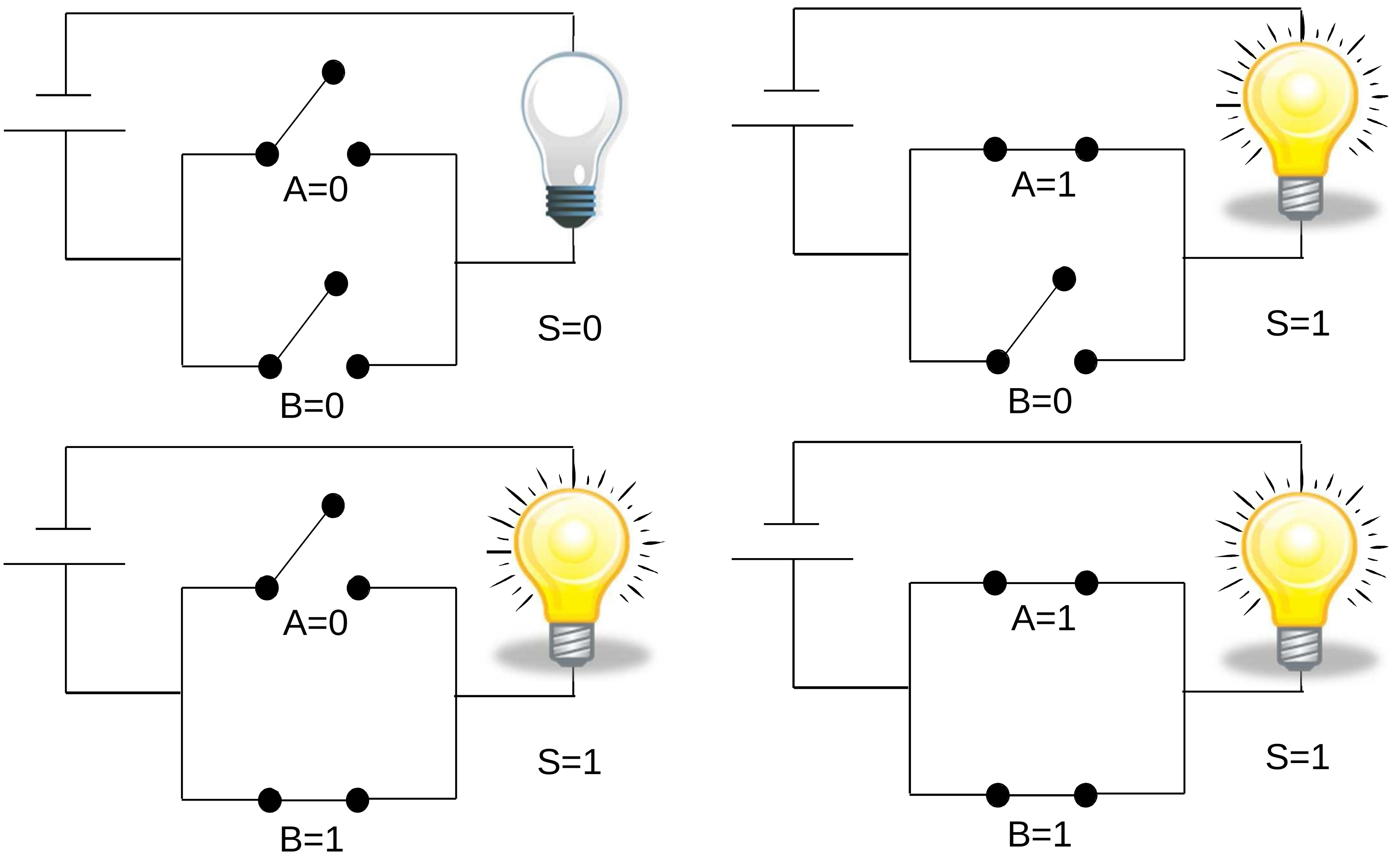


Função OU (OR)

- Executa a **soma** (**disjunção**) **booleana** de duas ou mais variáveis binárias.
- Por exemplo, assuma a convenção no circuito:
  - **Chave aberta = 0; Chave fechada = 1**
  - **Lâmpada apagada = 0; Lâmpada acesa = 1**



Função OU (OR)



Função OU (OR)

- ✓ *Se a chave **A está aberta (A=0)** e a chave **B aberta (B=0),** não haverá circulação de energia no circuito, logo **a lâmpada fica apagada (S=0).***

- ✓ *Se a chave **A está fechada (A=1)** e a chave **B aberta (B=0),** haverá circulação de energia no circuito e **a lâmpada fica acesa (S=1).***

- ✓ *Se a chave **A está aberta (A=0)** e a chave **B fechada (B=1),** haverá circulação de energia no circuito e **a lâmpada fica acesa (S=1).***

- ✓ *Se a chave **A está fechada (A=1)** e a chave **B fechada (B=1)**, haverá circulação de energia no circuito e **a lâmpada fica acesa (S=1).***

- ✓ **Conclusão:** observando todas as quatro situações possíveis (interpretações), é possível concluir que **a lâmpada fica acesa somente quando a chave A ou a chave B ou ambas estiverem fechadas.**

Função OU (OR)

Para representar a expressão

- ✓ S = A ou B

Adotaremos a representação

- ✓ S = A + B, onde se lê S = A ou B

Porém, existem notações alternativas

- ✓ S = A | B
- ✓ S = A ; B
- ✓ $S = A \vee B$

Tabela Verdade da Função OU (OR)

- Observe que, no **sistema de numeração binário**, a soma 1+1 = 10.

- Na **álgebra booleana**, 1+1 = 1, já que somente dois valores são permitidos (0 e 1).

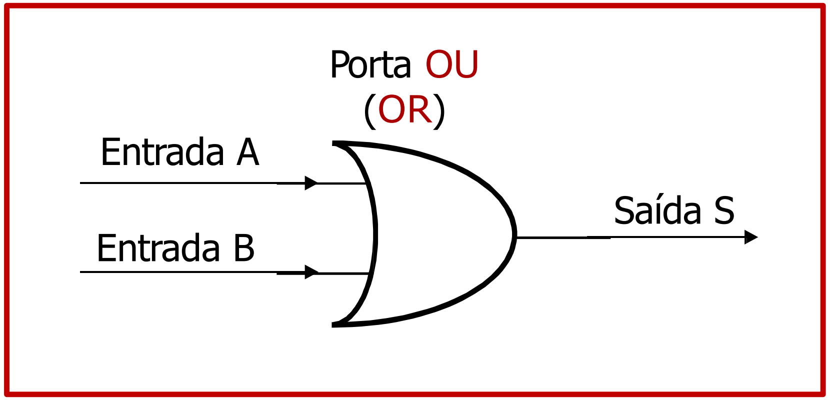
A	B	A + B
0	0	0
0	1	1
1	0	1
1	1	1

Porta Lógica OU (OR)

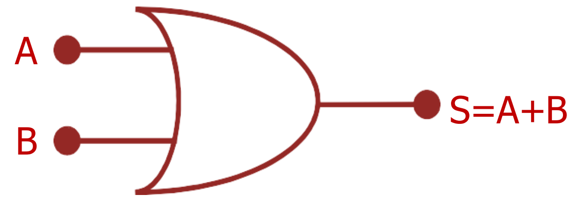
- A porta OU é um circuito que executa a função OU

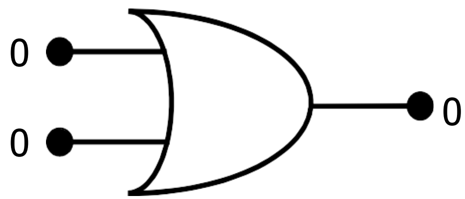
- A porta OU executa a tabela verdade da função OU
  - Portanto, a saída será 0 somente se ambas as entradas forem iguais a 0; nos demais casos, a saída será 1.

- Representação:

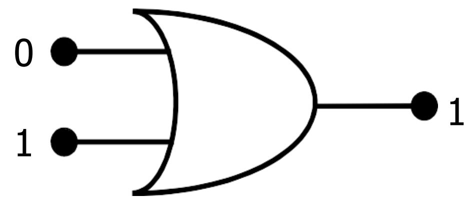


Porta Lógica OU (OR)

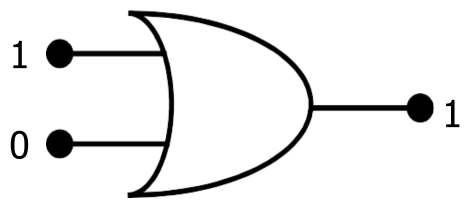




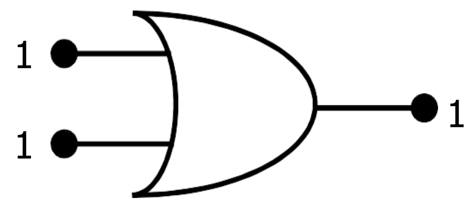
A	B	S=A+B
0	0	0
0	1	1
1	0	1
1	1	1



A	B	S=A+B
0	0	0
0	1	1
1	0	1
1	1	1



A	B	S=A+B
0	0	0
0	1	1
1	0	1
1	1	1



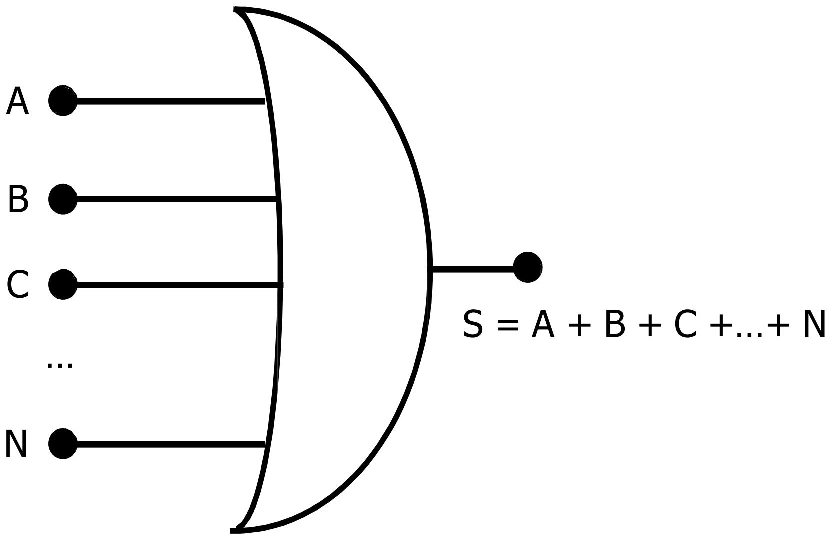
A	B	S=A+B
0	0	0
0	1	1
1	0	1
1	1	1

Porta Lógica OU (OR)

- É possível estender o conceito de uma porta OU para um número qualquer de variáveis de entrada.

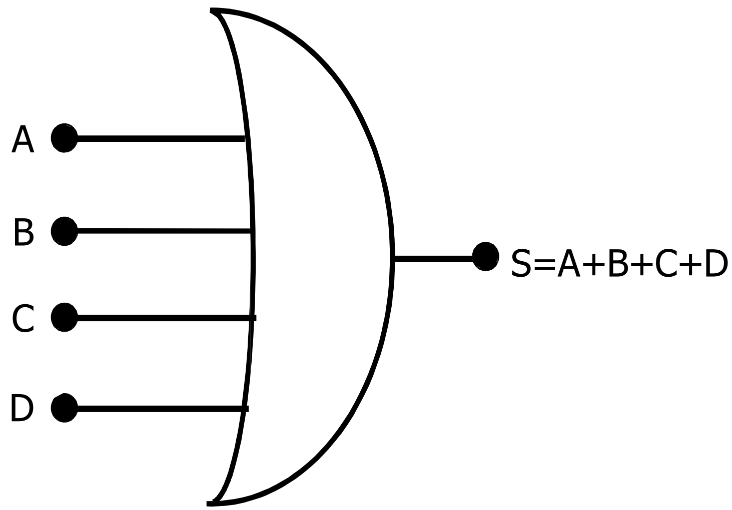
- Nesse caso, temos uma porta OU com N entradas e somente uma saída.

- A saída será 0 se e somente se as N entradas forem iguais a 0; nos demais casos, a saída será 1.



Porta Lógica OU (OR)

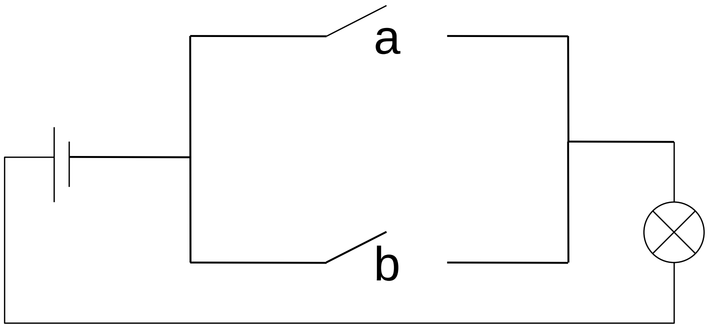
Por exemplo,
S = A + B + C + D



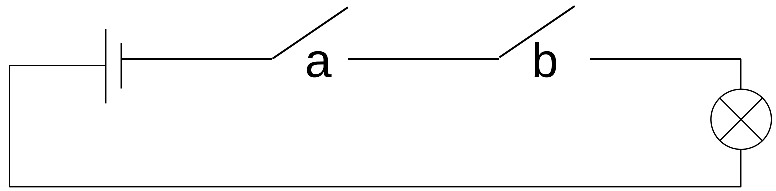
A	B	C	D	S
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

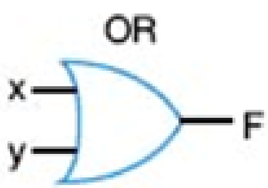
Respondendo... Qual a representação da passagem da corrente elétrica nos circuitos abaixo?

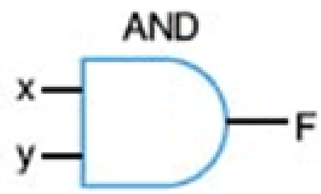
a + b



ab







a	b	a + b	ab
1	1	1	1
1	0	1	0
0	1	1	0
0	0	0	0

Função NÃO (NOT)

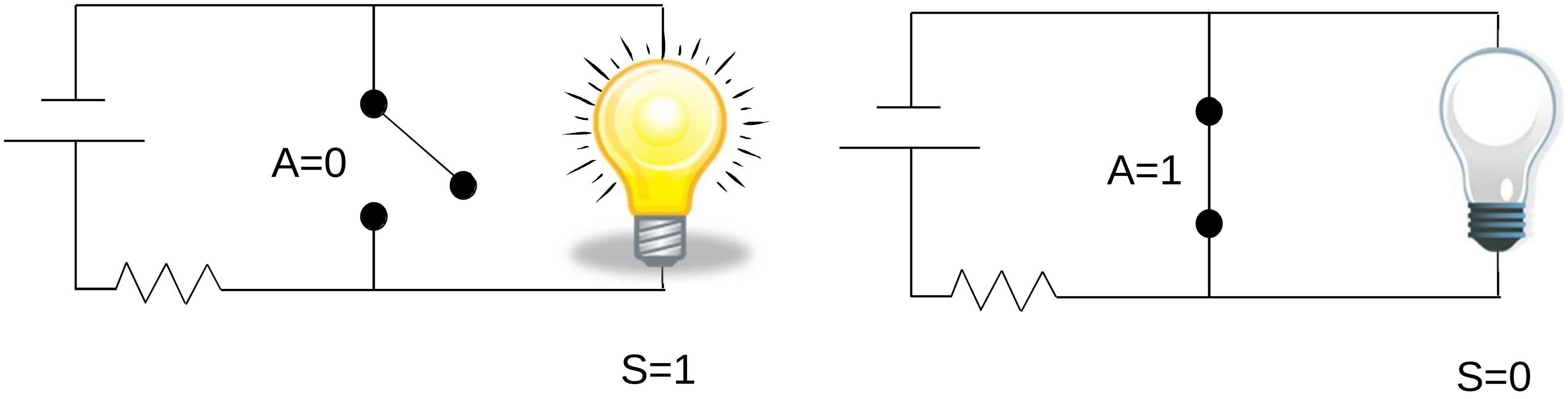
- Executa a **complemento** **(negação) booleana** de duas ou mais variáveis binárias.

  - Se a variável estiver em 0, o resultado da função é 1.
  - Se a variável estiver em 1, o resultado da função é 0.

- Essa função também é chamada de **inversora.**

Função NÃO (NOT)

- Usando as mesmas convenções dos circuitos anteriores, tem-se que:
  - Quando a chave A está aberta (A=0), passará corrente pela lâmpada e ela acenderá (S=1);
  - Quando a chave A está fechada (A=1), a lâmpada estará em curto-circuito e não passará corrente por ela, ficando apagada (S=0)



Função NÃO (NOT)

- Para representar a expressão
  - S = não A

- Adotaremos a representação
  - S = $\bar{A}$, onde se lê S = não A

- Notações alternativas
  - S = A’
  - S = ¬ A
  - S = Ã

Tabela verdade da função NÃO (NOT)

A	$\bar{A}$
0	1
1	0

Porta Lógica NÃO (NOT)

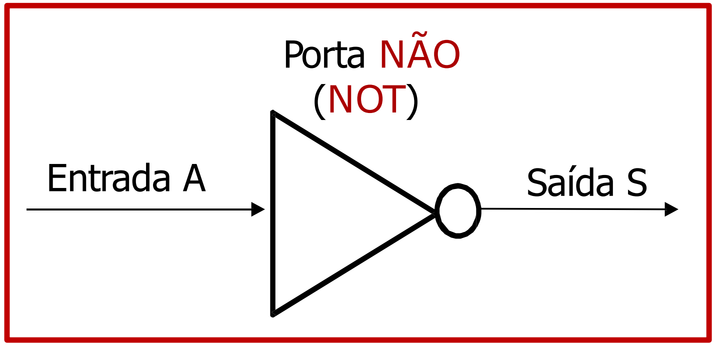
- A porta lógica NÃO, ou inversor, é o circuito que executa a função NÃO.

- O inversor executa a tabela verdade da função NÃO:

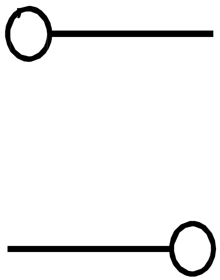
  Se a entrada for 0, a saída será 1;

  se a entrada for 1, a saída será 0.

Representação



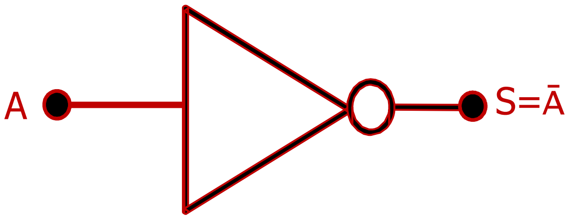
Alternativamente

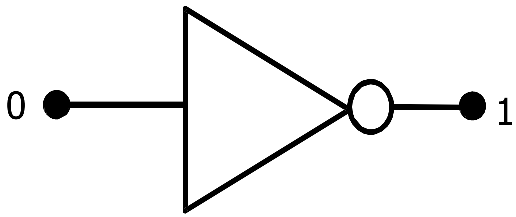


Após um bloco lógico

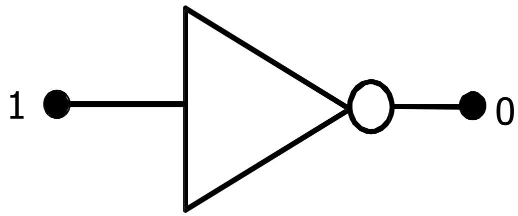
Antes de um bloco lógico

Porta Lógica NÃO (NOT)





A	S=Ā
0	1
1	0



A	S=Ā
0	1
1	0

Função "NÃO E" (NAND)

- Composição da função E com a função NÃO, ou seja, é a saída invertida da função E.

$S = \overline{(A.B)} = \overline{A.B}$
$= (A.B)'$
$= \neg(A.B)$

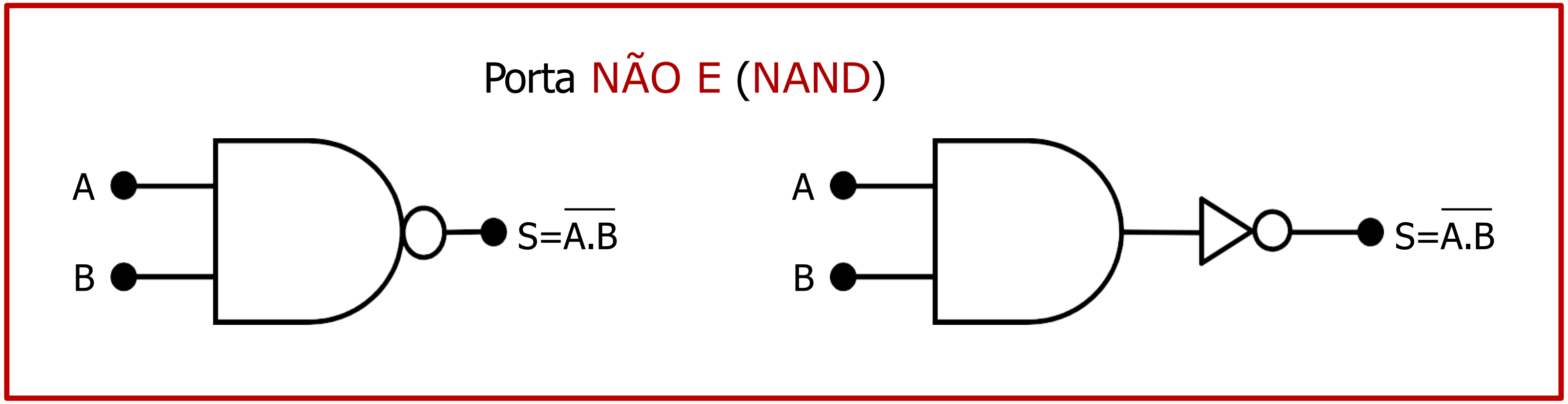
Tabela Verdade:

A	B	$\overline{A.B}$
0	0	1
0	1	1
1	0	1
1	1	0

Porta "NÃO E" (NAND)

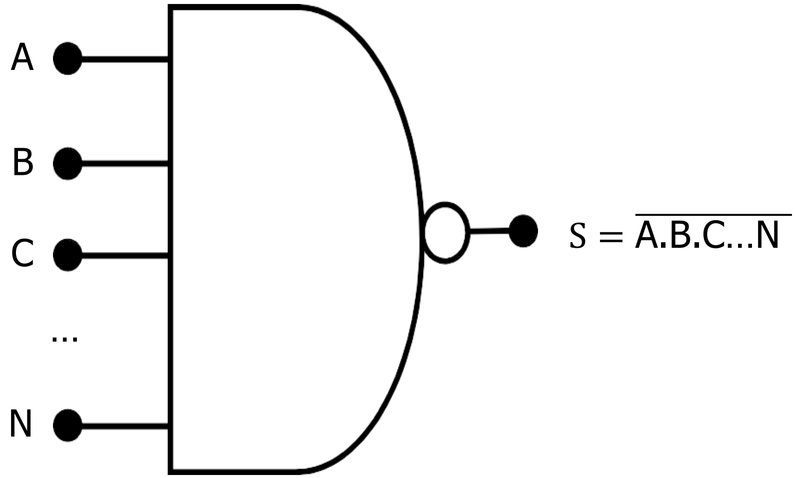
- A porta NÃO E (NE) é o bloco lógico que executa a função NÃO E, ou seja, sua tabela verdade

- Representação



Porta "NÃO E" (NAND)

- Como a porta E, a porta NÃO E pode ter duas ou mais entradas.
- Nesse caso, temos uma porta NÃO E com N entradas e somente uma saída.
- A saída será 0 se e somente se as N entradas forem iguais a 1; nos demais casos, a saída será 1.



Função " NÃO OU" (NOR)

- Composição da função OU com a função NÃO, ou seja, é a saída invertida da função OU.

S $= (\overline{A+B}) = \overline{A+B}$
$= (A+B)'$
$= \neg(A+B)$

Tabela Verdade:

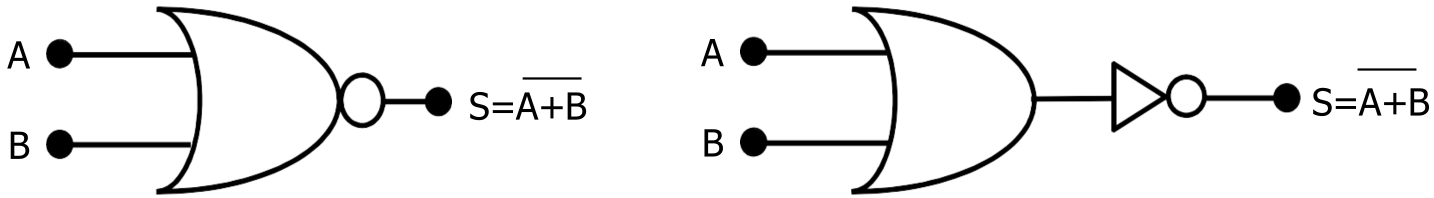
A	B	$\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

Porta "NÃO OU" (NOR)

- A porta NÃO OU (NOU) é o bloco lógico que executa a função NÃO OU, ou seja, sua tabela verdade

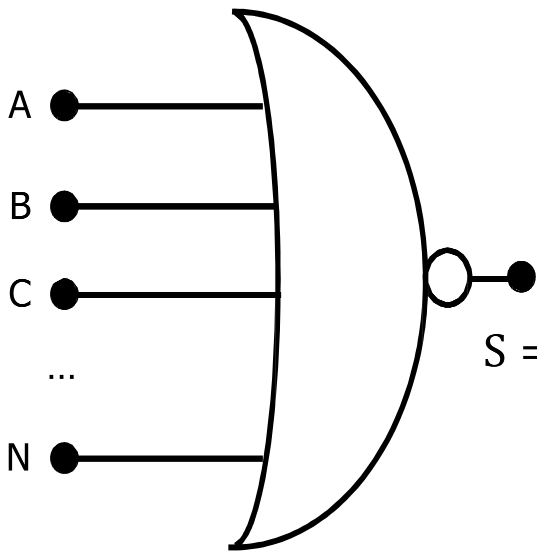
- Representação

Porta NÃO OU (NOR)



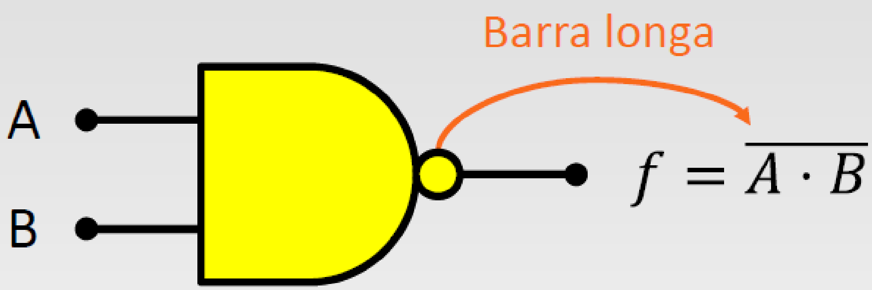
Porta NÃO OU (NOR)

- Como a porta OU, a porta NÃO OU pode ter duas ou mais entradas.
- Nesse caso, temos uma porta NÃO OU com N entradas e somente uma saída.
- A saída será 1 se e somente se as N entradas forem iguais a 0; nos demais casos, a saída será 0.

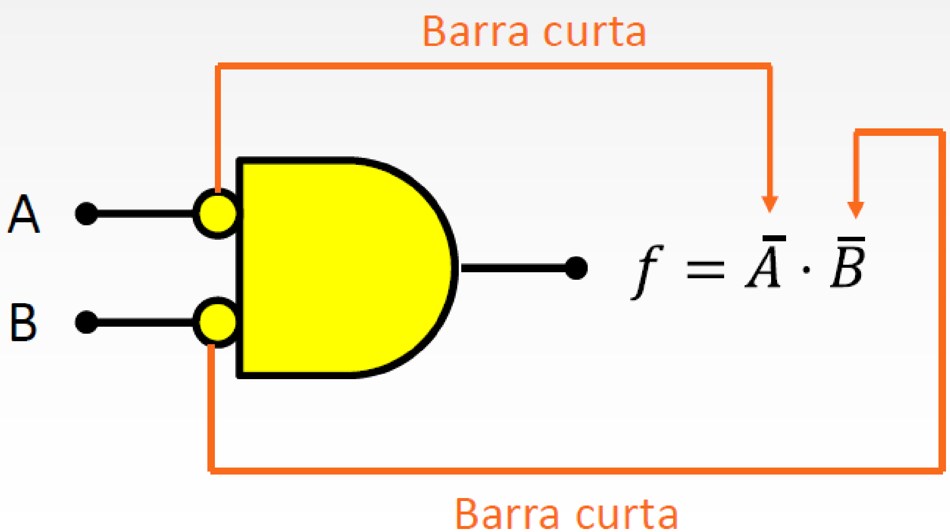


$$S = \overline{A+B+C+...+N}$$

Não Confunda!



A	B	$\overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0



A	B	$\bar{A} \cdot \bar{B}$
0	0	1
0	1	0
1	0	0
1	1	0

Portas NAND e NOR

- As portas NOR e NAND são chamadas universais, pois qualquer função lógica pode ser construída usando-se apenas portas NAND ou apenas portas NOR.

Função OU Exclusivo (XOR)

- A função OU Exclusivo :

✓ fornece *1 na saída quando as entradas forem diferentes entre si; e*

✓ fornece *0 caso contrário.*

✓ *Implementa a SOMA de binários, desconsiderando o Vai-Um.*

Tabela verdade:

A	B	$S=A\oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

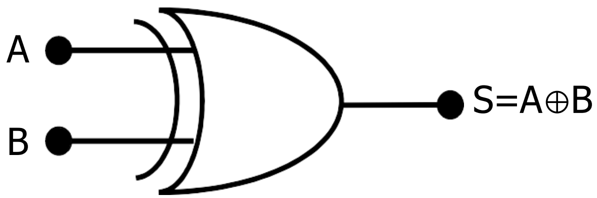
- $S = A \oplus B$

  $= \overline{A}B + A\overline{B}$.

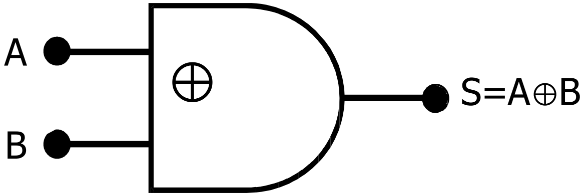
Porta OU Exclusivo (XOR) como Bloco Básico

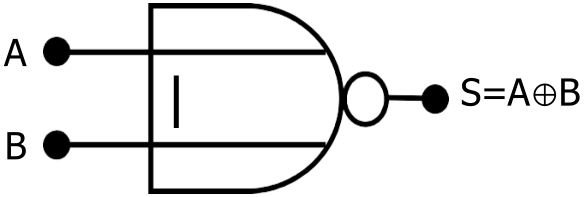
Simbologia adotada:

Porta XOR



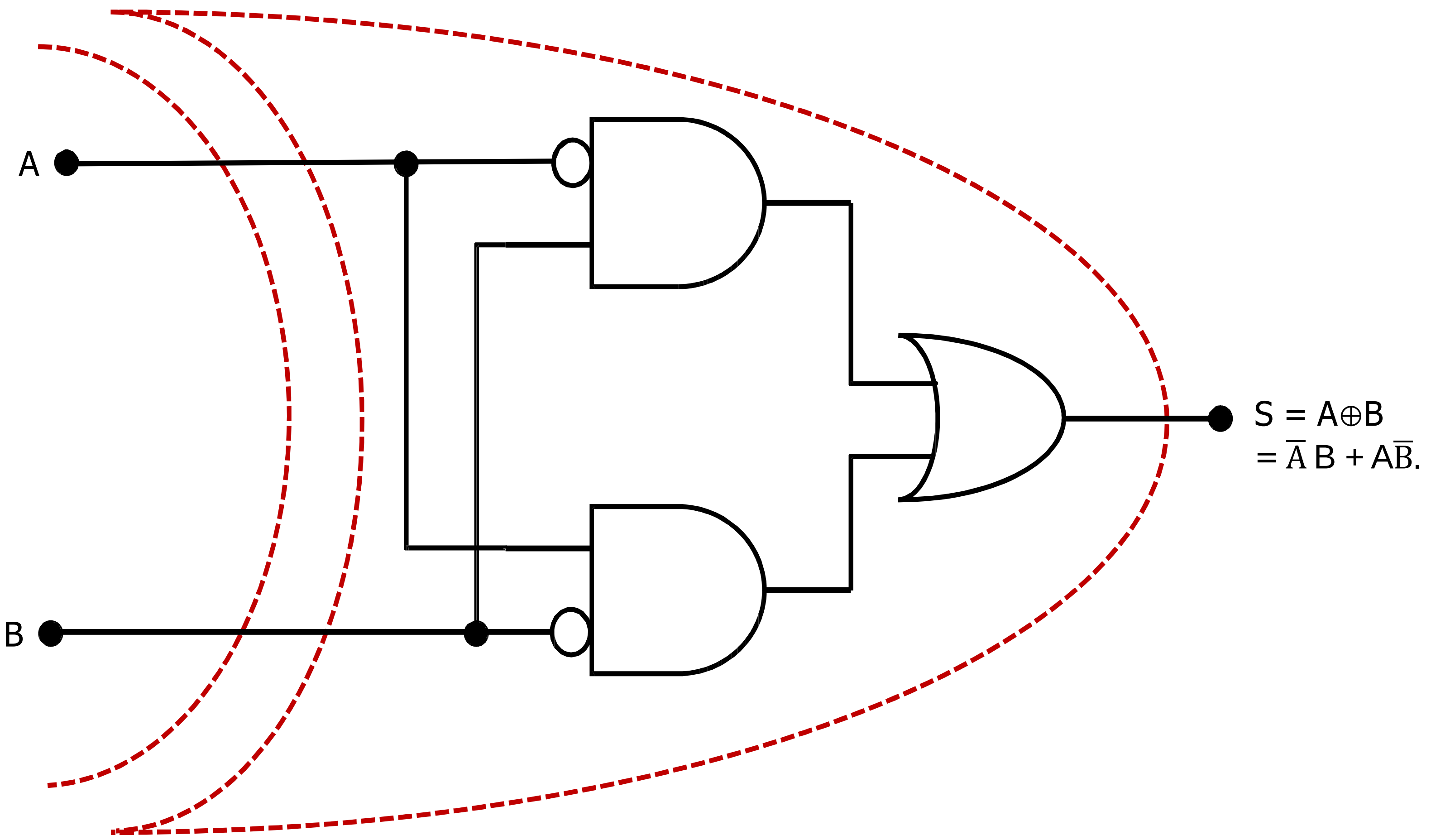
Outros símbolos utilizados





48

Porta OU Exclusivo (XOR) como Circuito Combinacional



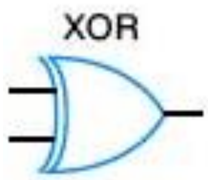
Função Coincidência (XNOR)

- **XNOR (NOR Exclusivo)** = Oposto de XOR:

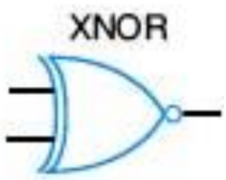
✓ função de coincidência (equivalência);

✓ fornece 1 na saída quando as entradas forem iguais entre si; e fornece 0 caso contrário.

$F = xy' + x'y$



x	y	F
0	0	0
0	1	1
1	0	1
1	1	0



x	y	F
0	0	1
0	1	0
1	0	0
1	1	1

$F = x'y'+xy$

Atenção!! Portas XOR e XNOR

No caso de 3 variáveis, as expressões são:

$$X = A \oplus B \oplus C$$

$$X = \overline{A \oplus B \oplus C}$$

Para montar a tabela da verdade deve-se primeiramente efetuar as operações entre 2 das variáveis e, com o resultado obtido, efetuar a operação com a terceira variável. Este processo se deve ao fato de as funções ***OU EXCLUSIVO*** *e* ***COINCIDÊNCIA*** *não serem válidas para mais de 2 variáveis de entrada.*

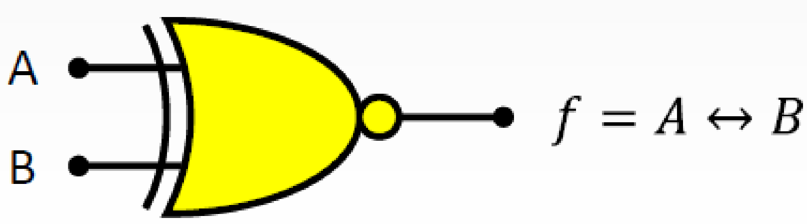
Se e Somente Se...

Definição	Verifica se uma expressão condicional do tipo "Se e somente se" é verdadeira
Representação	$A \leftrightarrow B$
Expressão equivalente	$(A \rightarrow B) \cdot (B \rightarrow A)$

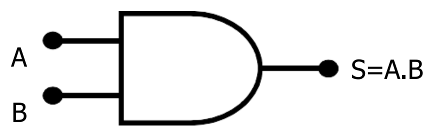
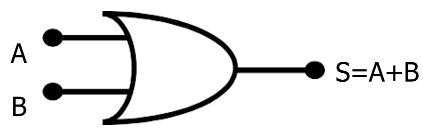
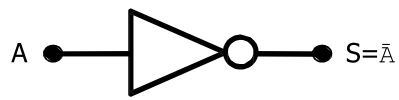
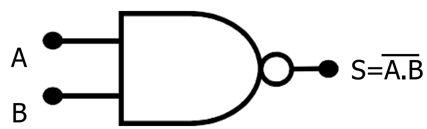
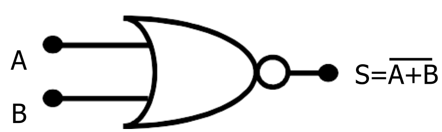
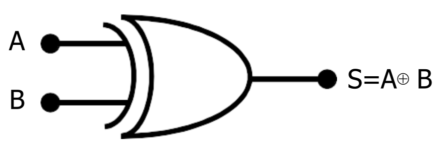
Tabela verdade de $A \leftrightarrow B$

A	B	$A \leftrightarrow B$
0	0	1
0	1	0
1	0	0
1	1	1

Porta Lógica XNOR:



Resumo dos Blocos Lógicos Básicos

Nome	Símbolo Gráfico	Função Algébrica	Tabela Verdade
E (AND)		S=A.B S=AB	A B | S=A.B 0 0 | 0 0 1 | 0 1 0 | 0 1 1 | 1
OU (OR)		S=A+B	A B | S=A+B 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 1
NÃO (NOT) Inversor		S=$\bar{A}$ S=A' S= ¬ A	A | S=$\bar{A}$ 0 | 1 1 | 0
NE (NAND)		S=$\overline{A.B}$ S=(A.B)' S= ¬(A.B)	A B | S=$\overline{A.B}$ 0 0 | 1 0 1 | 1 1 0 | 1 1 1 | 0
NOU (NOR)		S=$\overline{A+B}$ S=(A+B)' S=¬(A+B)	A B | S=$\overline{A+B}$ 0 0 | 1 0 1 | 0 1 0 | 0 1 1 | 0
XOR		S=A⊕B S=$\bar{A}$ B + A$\bar{B}$	A B | S=A⊕ B 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 0

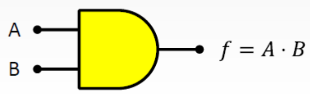
Resumo das Portas Lógicas

Definição	Resulta em verdade se e somente se todas as variáveis de entrada forem verdadeiras
Representação	$A \cdot B$
Representações alternativas	AB ou $A \wedge B$

Tabela verdade de $A \cdot B$

A	B	$A \cdot B$
0	0	0
0	1	0
1	0	0
1	1	1

Porta Lógica AND:

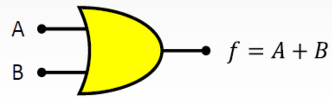


Definição	Resulta em verdade se e somente se pelo menos uma das variáveis de entrada for verdadeira
Representação	$A + B$
Representações alternativas	$A \vee B$

Tabela verdade de $A + B$

A	B	$A + B$
0	0	0
0	1	1
1	0	1
1	1	1

Porta Lógica OR:

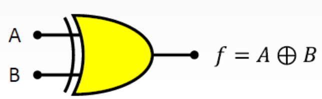


Definição	Resulta em verdade se e somente se apenas uma das duas entradas for verdadeira
Representação	$A \oplus B$
Expressão equivalente	$\bar{A} \cdot B + A \cdot \bar{B}$

Tabela verdade de $A \oplus B$

A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

Porta Lógica XOR:

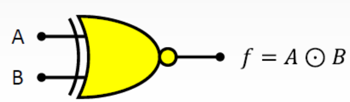


Definição	Resulta em verdade se e somente se todas as entradas tiverem o mesmo valor lógico
Representação	$A \odot B$
Expressão equivalente	$\bar{A} \cdot \bar{B} + A \cdot B$

Tabela verdade de $A \odot B$

A	B	$A \odot B$
0	0	1
0	1	0
1	0	0
1	1	1

Porta Lógica XNOR:

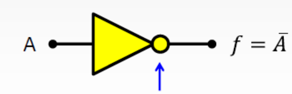


Definição	Troca o valor lógico da variável pelo seu complemento
Representação	$\bar{X}$
Representações alternativas	X' ou $\neg X$

Tabela verdade de $\bar{A}$

A	$\bar{A}$
0	1
1	0

Porta Lógica NOT:

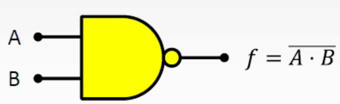


A presença de um pequeno círculo sempre indica inversão.

- Implementa o complemento da conjunção.
- Contração das palavras NOT e AND.

Entradas		Saída
A	B	$\overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

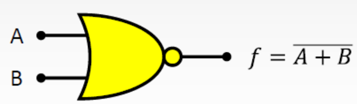
Porta Lógica NAND:



- Implementa o complemento da disjunção.
- Contração das palavras NOT e OR.

Entradas		Saída
A	B	$\overline{A + B}$
0	0	1
0	1	0
1	0	0
1	1	0

Porta Lógica NOR:



Símbolos Lógicos do Padrão IEEE/ANSI

- O Padrão IEEE/ANSI ainda não foi amplamente aceito para uso na área digital.

- A maioria dos data book de CI digital inclui os símbolos tradicionais e os símbolos IEEE/ANSI.

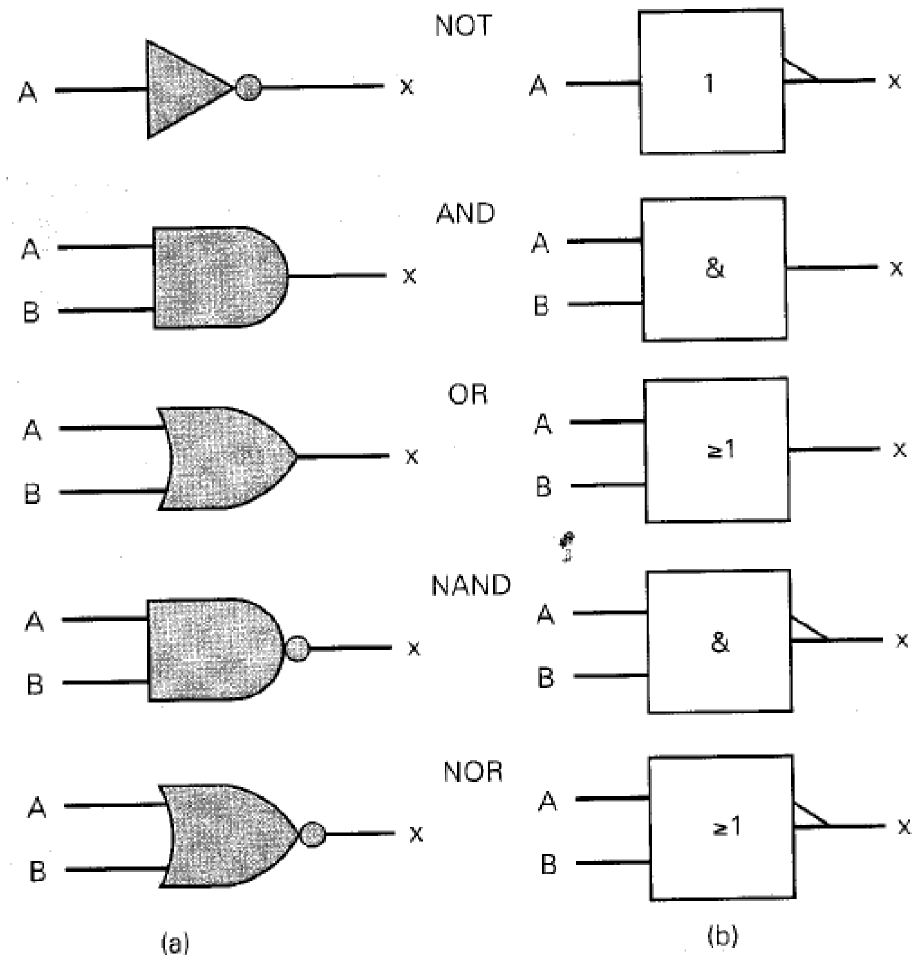
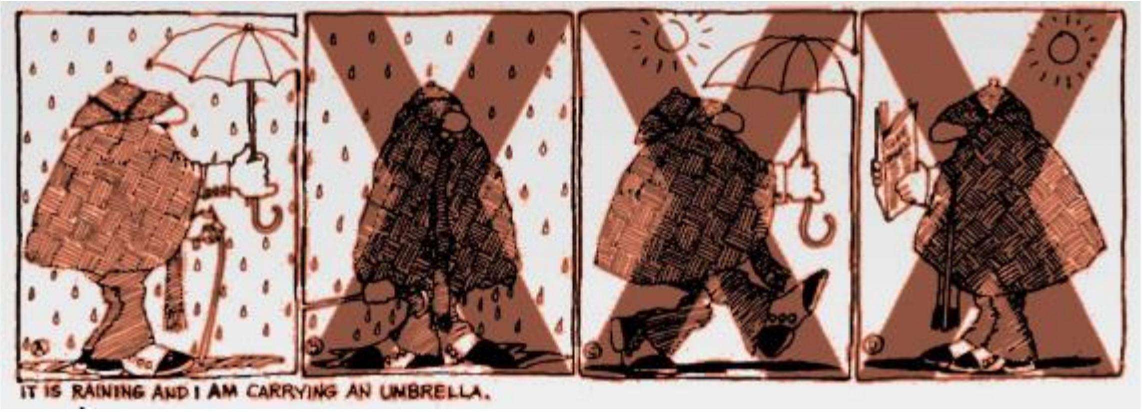


FIGURA 3.41 Símbolos lógicos-padrão: (a) tradicional; (b) IEEE/ANSI.

Exercícios



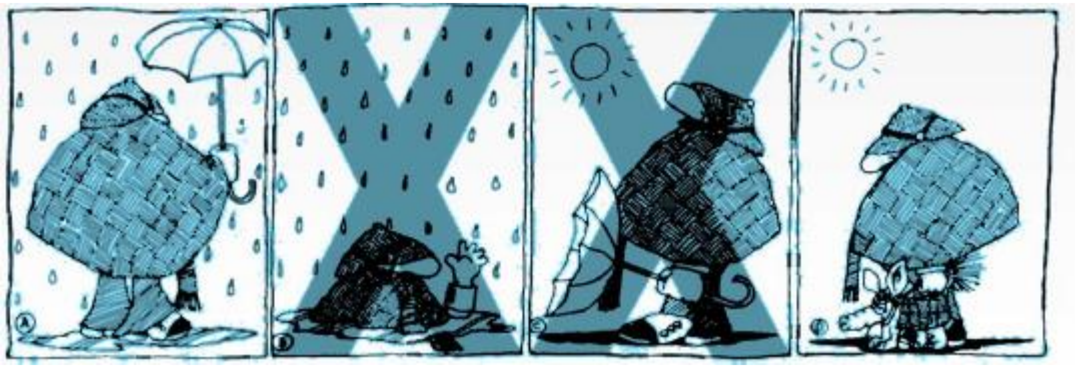
Está chovendo e estou carregando um guarda-chuva

A = está chovendo
B = estou carregando guarda-chuva

Está chovendo ou estou carregando um guarda-chuva

Exercícios

Ou está chovendo, ou estou carregando um guarda-chuva



Ou não está chovendo ou não estou carregando um guarda-chuva

Operação Implicação

Definição	Verifica se uma expressão condicional do tipo "Se X então Y" é verdadeira
Representação	$A \rightarrow B$
Expressão equivalente	$\bar{A} + B$

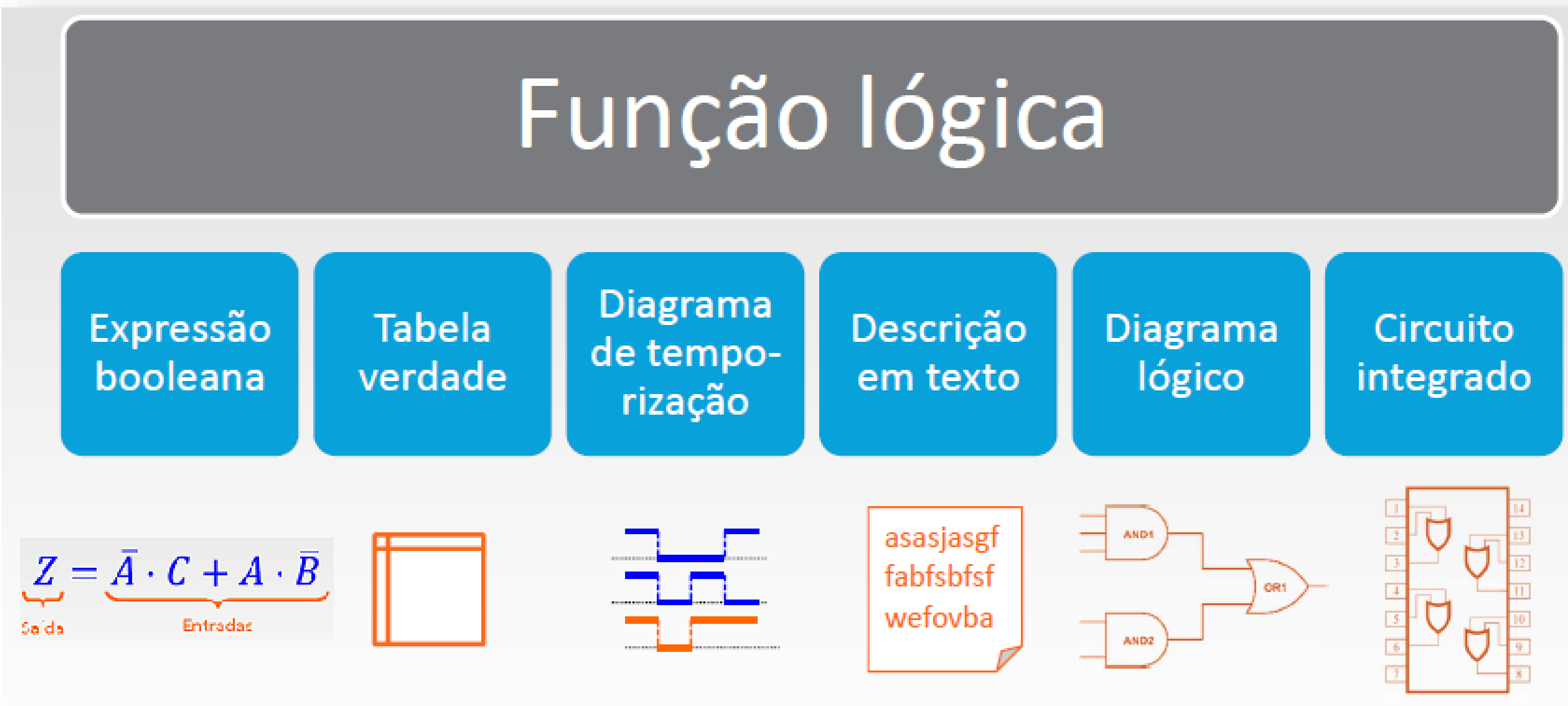
Tabela verdade de $A \rightarrow B$

A	B	$A \rightarrow B$
0	0	1
0	1	1
1	0	0
1	1	1

Não tem porta lógica associada.

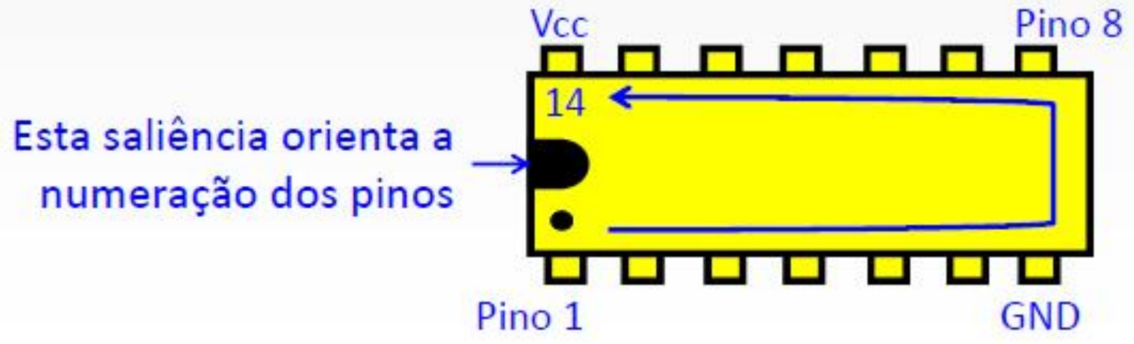
Formas de Representação de uma Função Lógica

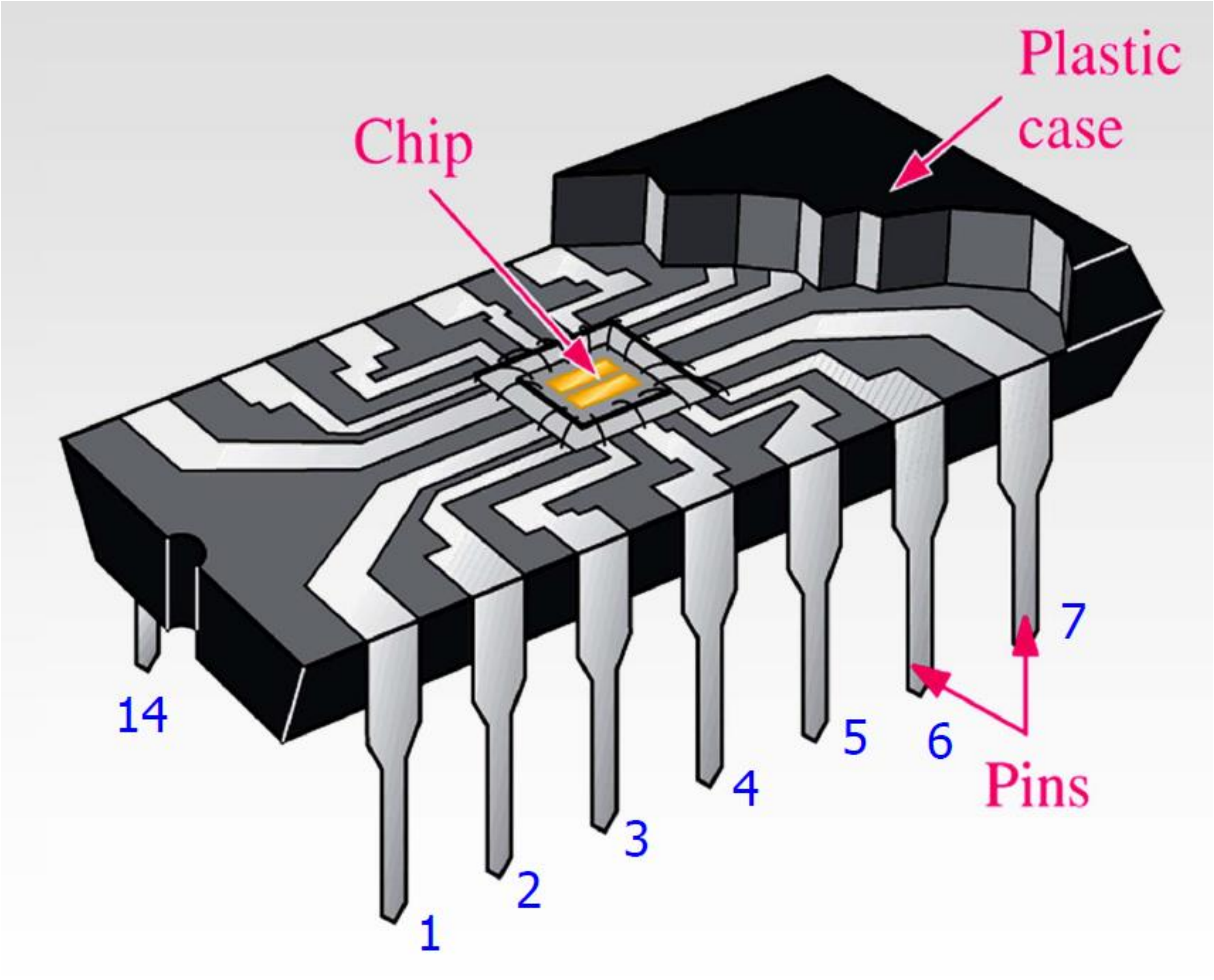
Formas de Representação de uma Função Lógica



Chips Digitais

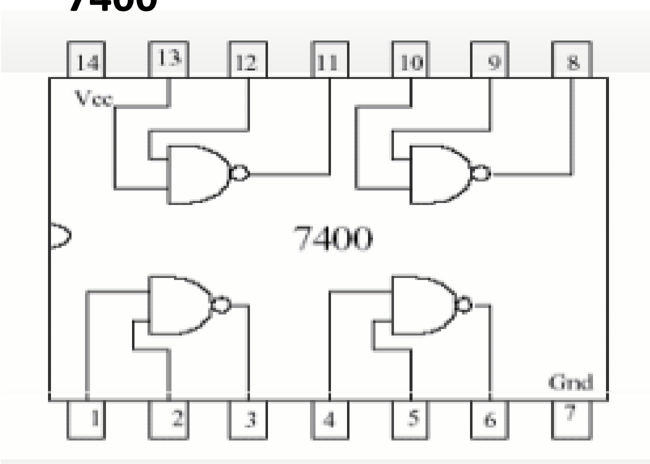
- As portas lógicas geralmente estão disponíveis em chips ou circuitos integrados (CI).
- Os modelos dos chips são identificados com por padrões de numeração.
  - Exemplos: 7404, 7408, 7432.
- Chips lógicos básicos geralmente têm 14 pinos.
- Dois pinos são usados para alimentação:
  - Vcc: +5 volts
  - GND (ground): terra



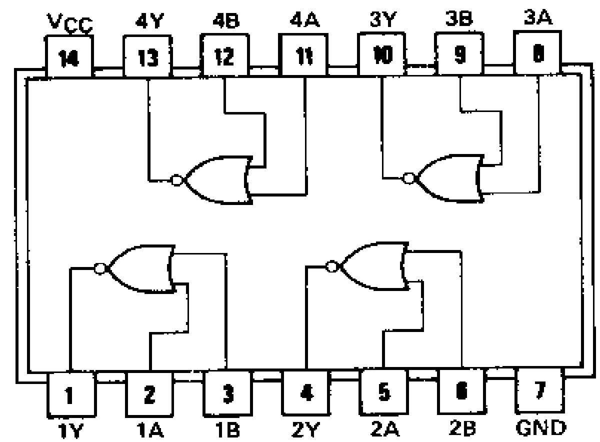

Plastic case
Chip
14
1
2
3
4
5
6
7
Pins

Kits de CI’s Lógicos

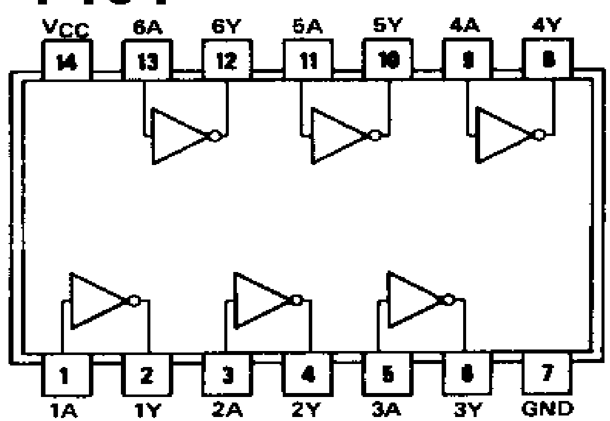
7400



7402

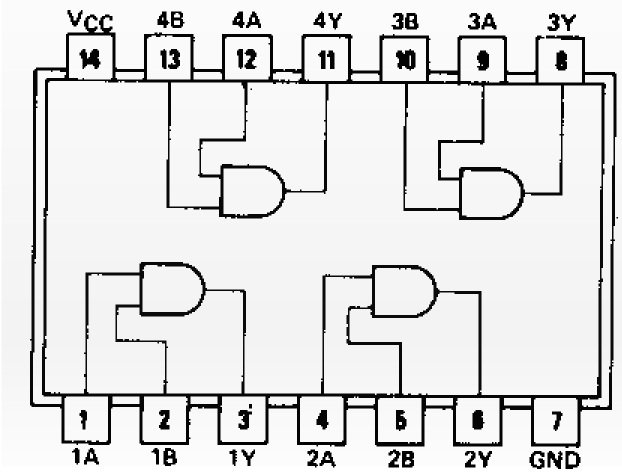


7404

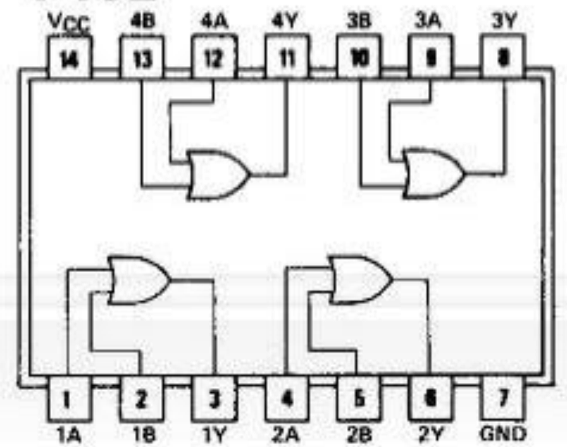


Kits de CI’s Lógicos

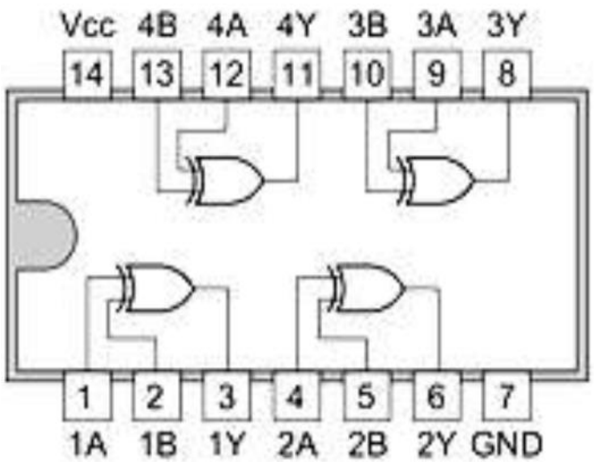
7408



7432



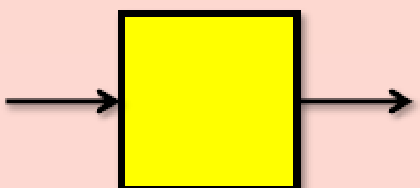
7486



Circuitos Lógicos

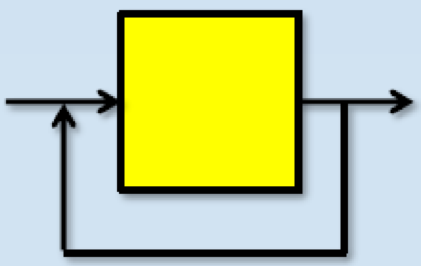
Circuitos combinatórios

suas saídas dependem unicamente de suas entradas



Circuitos sequenciais

estado lógico atual depende das entradas atuais e do estado lógico anterior



Circuito Combinacional

- Um circuito combinacional é um sistema digital no qual o valor da saída em qualquer instante depende somente do valor da entrada neste mesmo instante (e não de valores anteriores).

$$z = F(x)$$

Diagrama de Temporização

- Um diagrama de temporização é uma representação de um conjunto de sinais no domínio do tempo.
- Fornece uma descrição geral das relações de tempo, ajudando a localizar e diagnosticar problemas.

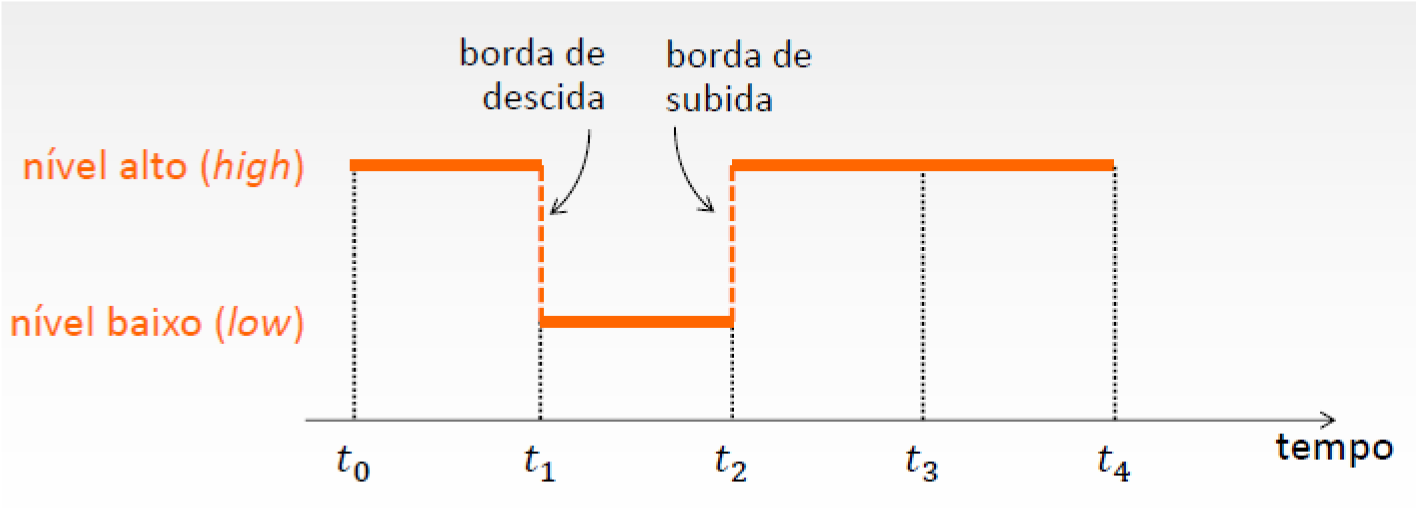
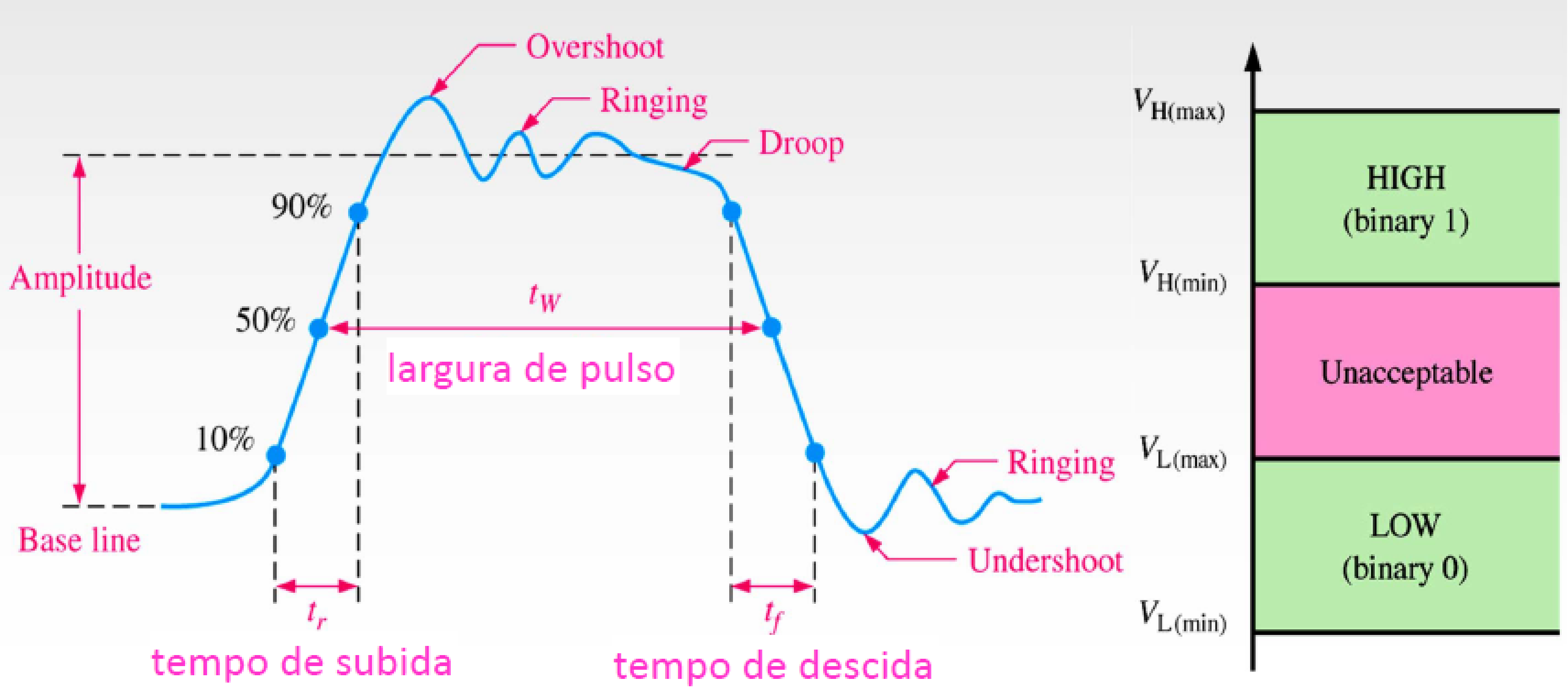


Diagrama de Temporização

Idealmente	Na prática
As transições (bordas) do sinal de saída ocorrem <u>simultaneamente</u> às transições do sinal de entrada	Existe um pequeno <u>atraso</u> entre a transição de entrada e a transição de saída correspondente

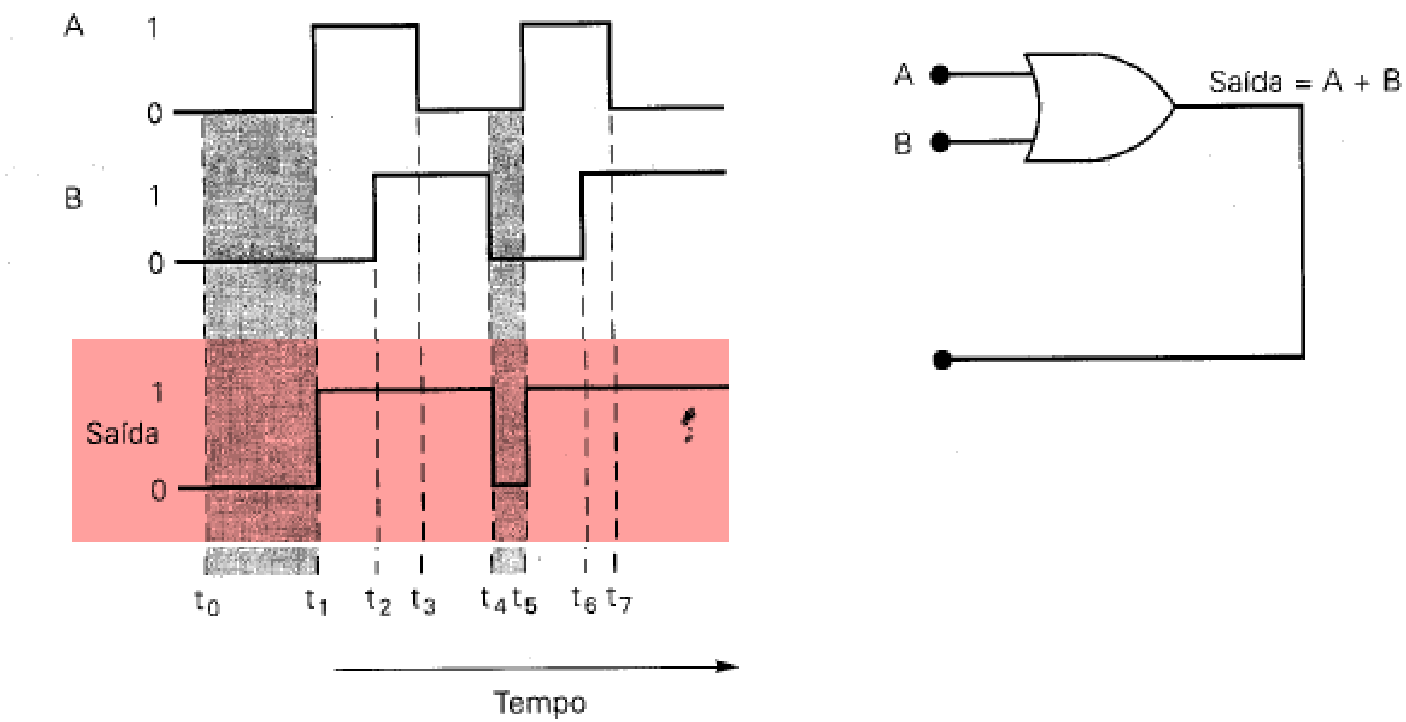
Formas de Onda

- Na prática, nunca ocorre transição instantânea de tensão.
- Porém, pode-se abstrair esse fato quando se considera o nível lógico de um sistema.

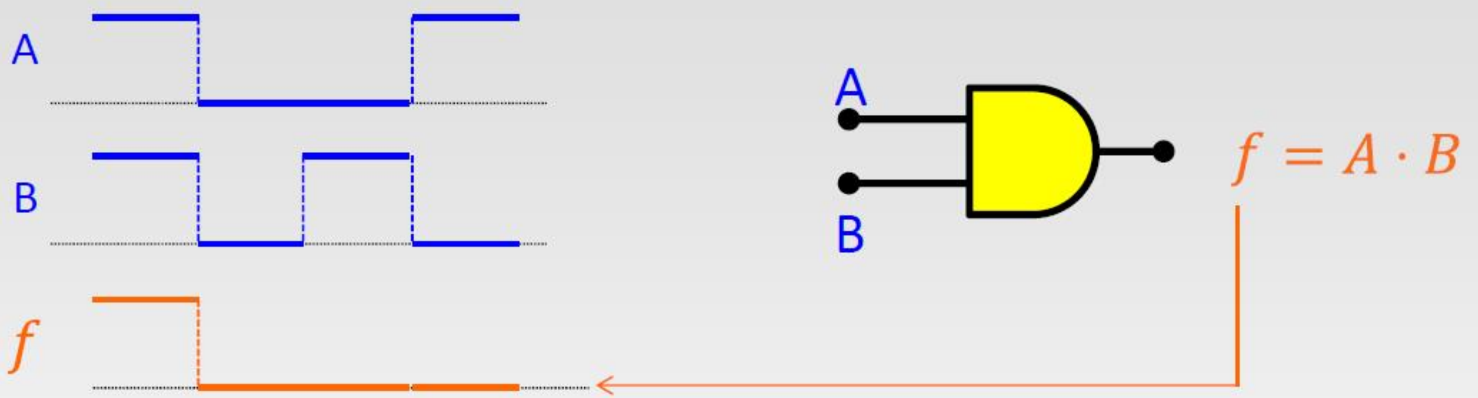


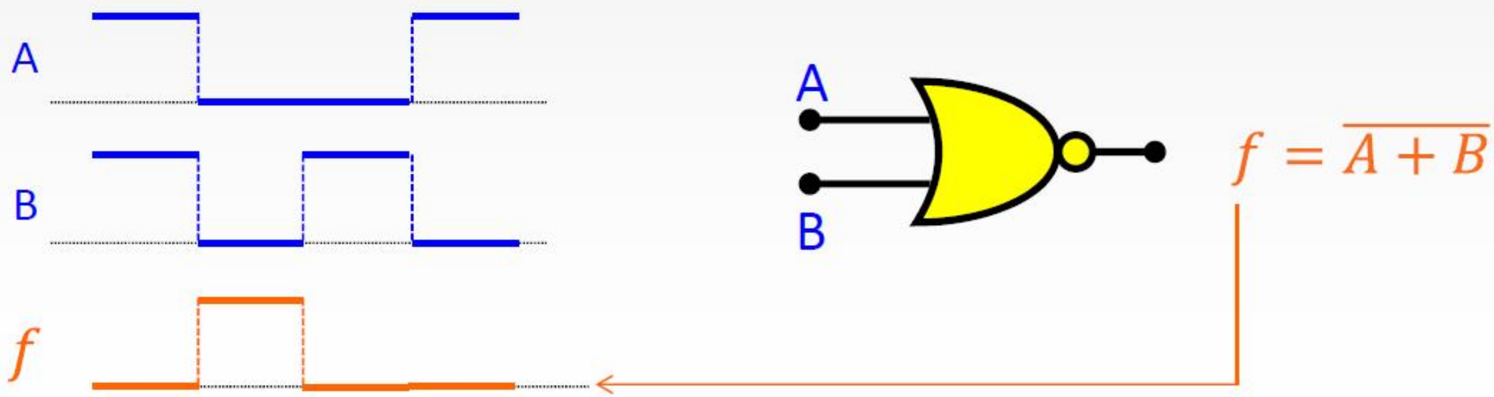
Representação no Tempo

Determine a saída da porta OR na Figura 3.5. As entradas A e B da porta OR variam de acordo com o diagrama de tempo mostrado. Por exemplo, a entrada A começa no nível BAIXO no instante t_0, muda para ALTO em t_1, volta para BAIXO em t_3 e assim por diante.



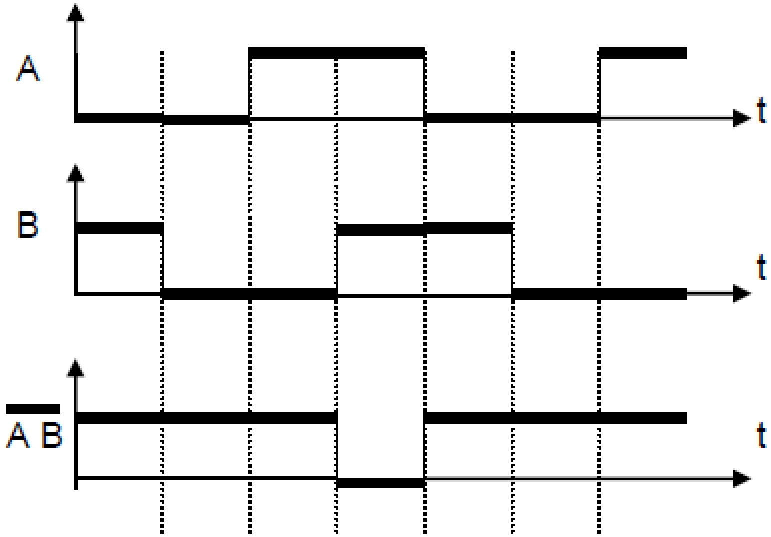
Diagramas de Temporização
:: Exemplos

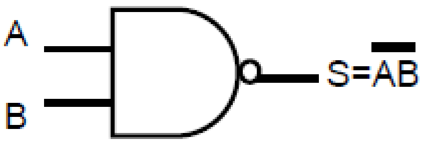




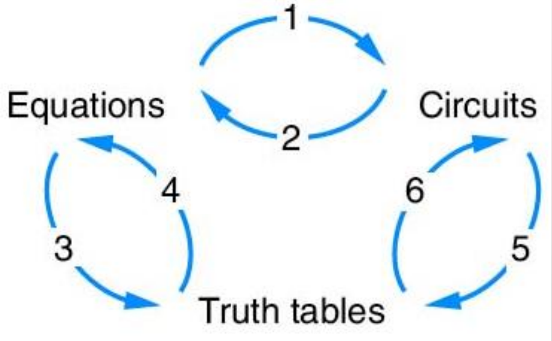
Representação no Tempo

A figura abaixo mostra um *Diagrama de Tempo* para A e B variando em relação ao tempo e a correspondente variação da saída $\overline{A\,B}$





Correspondência entre Expressões, Circuitos e Tabelas Verdade



Correspondência entre expressões, circuitos e tabelas verdade

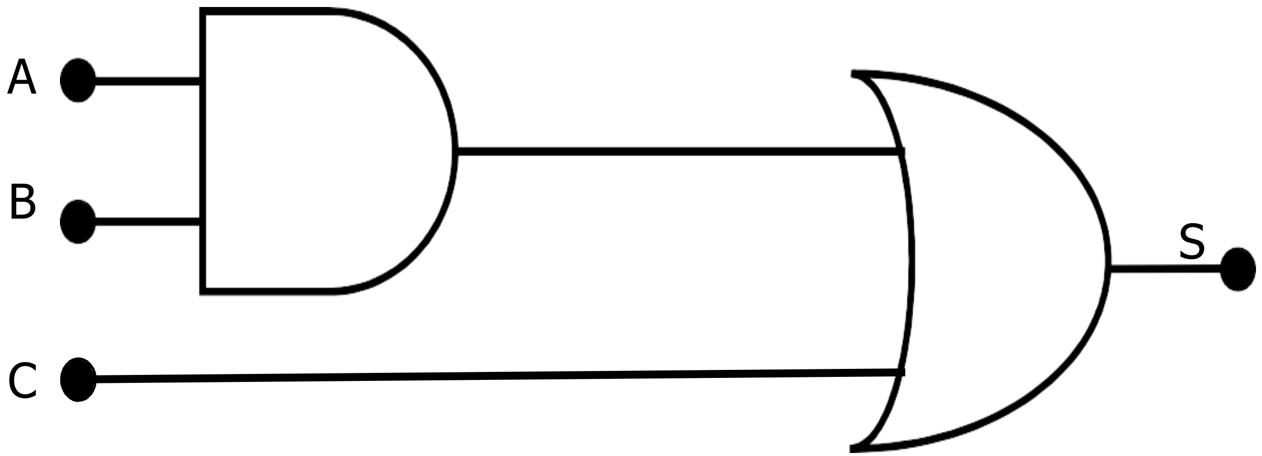
- Todo circuito lógico executa uma expressão booleana.

- Um circuito, por mais complexo que seja, é composto pela interligação dos blocos lógicos básicos.

- Veremos, a seguir, **como obter as expressões booleanas geradas por um circuito lógico combinacional**.

Expressões Booleanas Geradas por Circuitos Lógicos

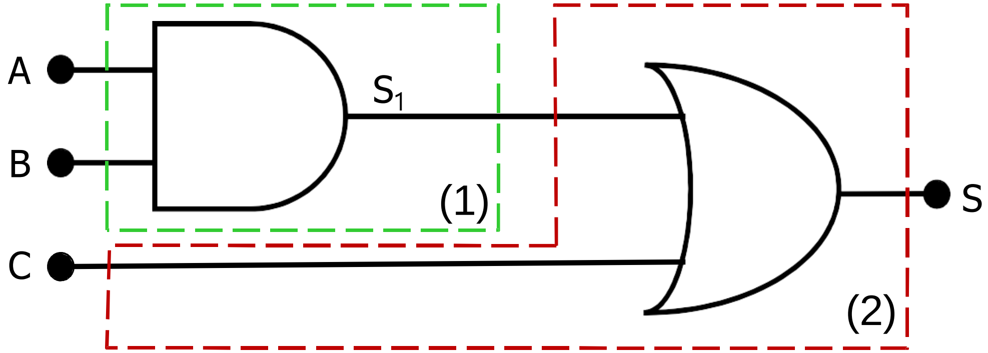
Seja o circuito:



Expressões Booleanas Geradas por Circuitos Lógicos

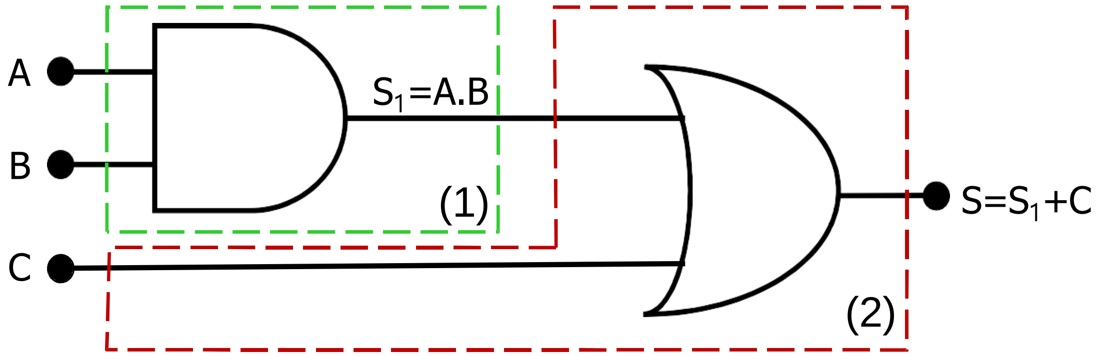
Vamos dividi-lo em duas partes (1) e (2):

- No circuito (1), a saída S_1 contém o produto A.B, já que o bloco é uma porta E
- Portanto, S_1 = A.B



Expressões Booleanas Geradas por Circuitos Lógicos

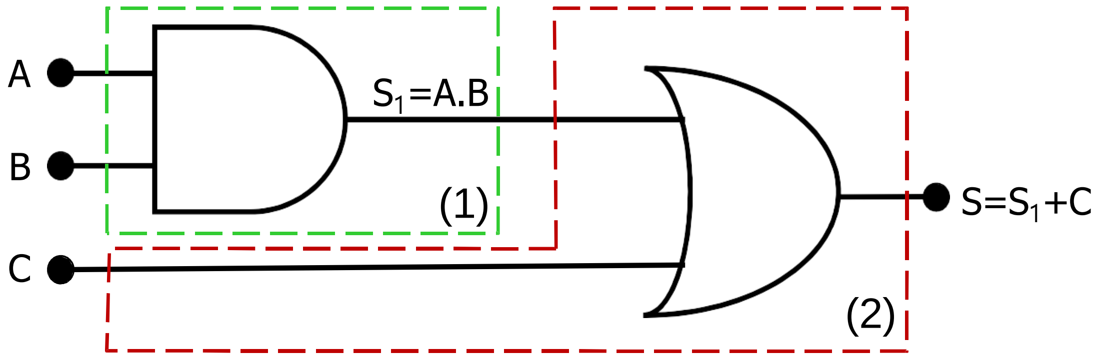
- No circuito (2), note que a saída S_1 é utilizada como uma das entradas da porta OU.
- A outra entrada da porta OU corresponde à variável C, o que nos leva à:
  - $S = S_1 + C$



Expressões Booleanas Geradas por Circuitos Lógicos

- Para obter a expressão final em relação às entradas A, B e C basta substituir a expressão S1 na expressão de S, ou seja:
  - (1) S1 = A.B
  - (2) S = S1 + C

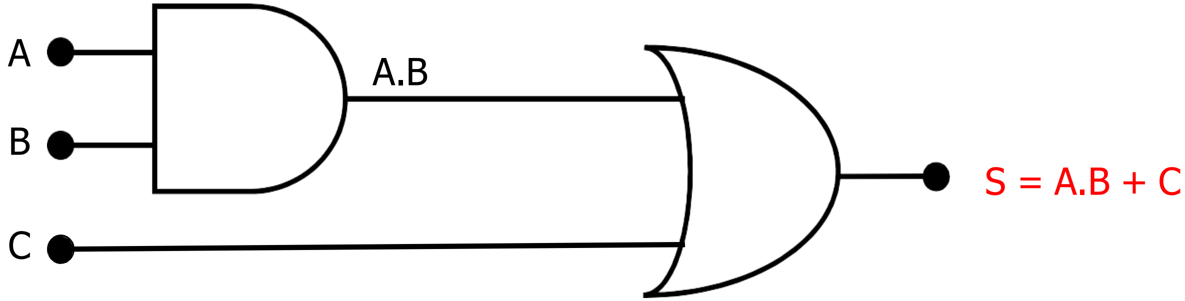
- Obtém-se S = S1 + C → S = (A.B) + C



Expressões Booleanas Geradas por Circuitos Lógicos

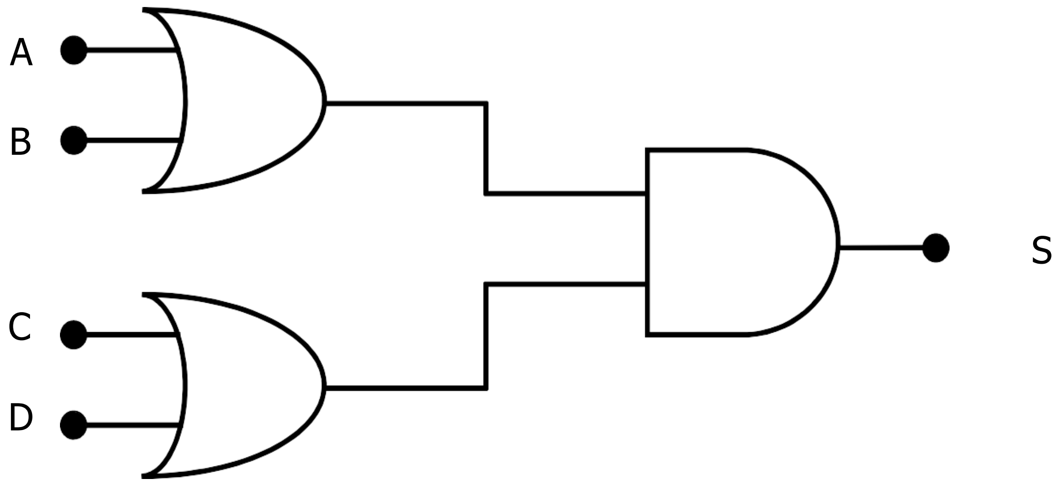
- Portanto, a expressão que o circuito executa é:

$$S = (A.B) + C = A.B + C$$



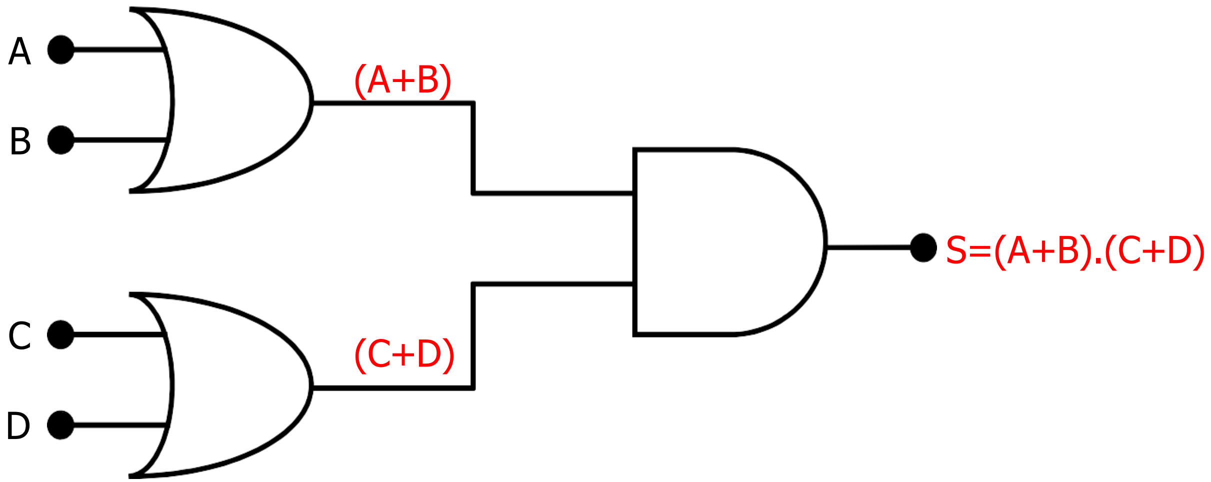
Expressões Booleanas Geradas por Circuitos Lógicos

<u>EXEMPLO 1:</u> Escreva a expressão booleana executada pelo circuito:



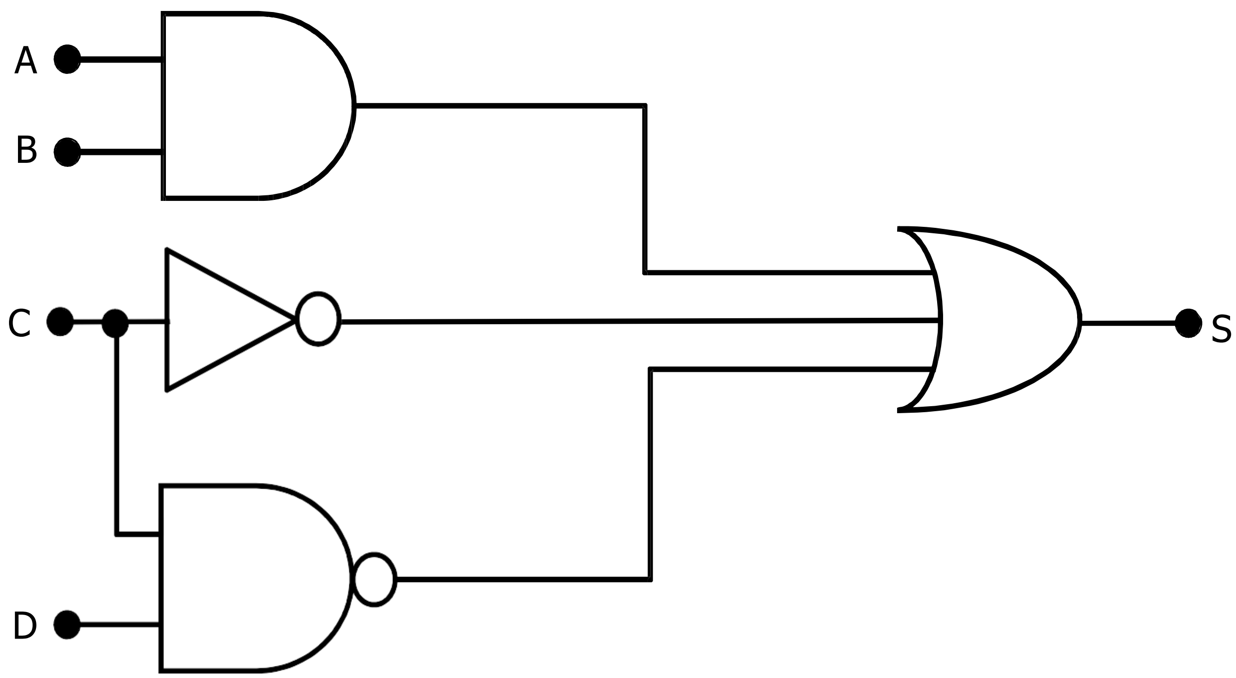
Expressões Booleanas Geradas por Circuitos Lógicos

<u>**EXEMPLO 1:**</u> Escreva a expressão booleana executada pelo circuito:



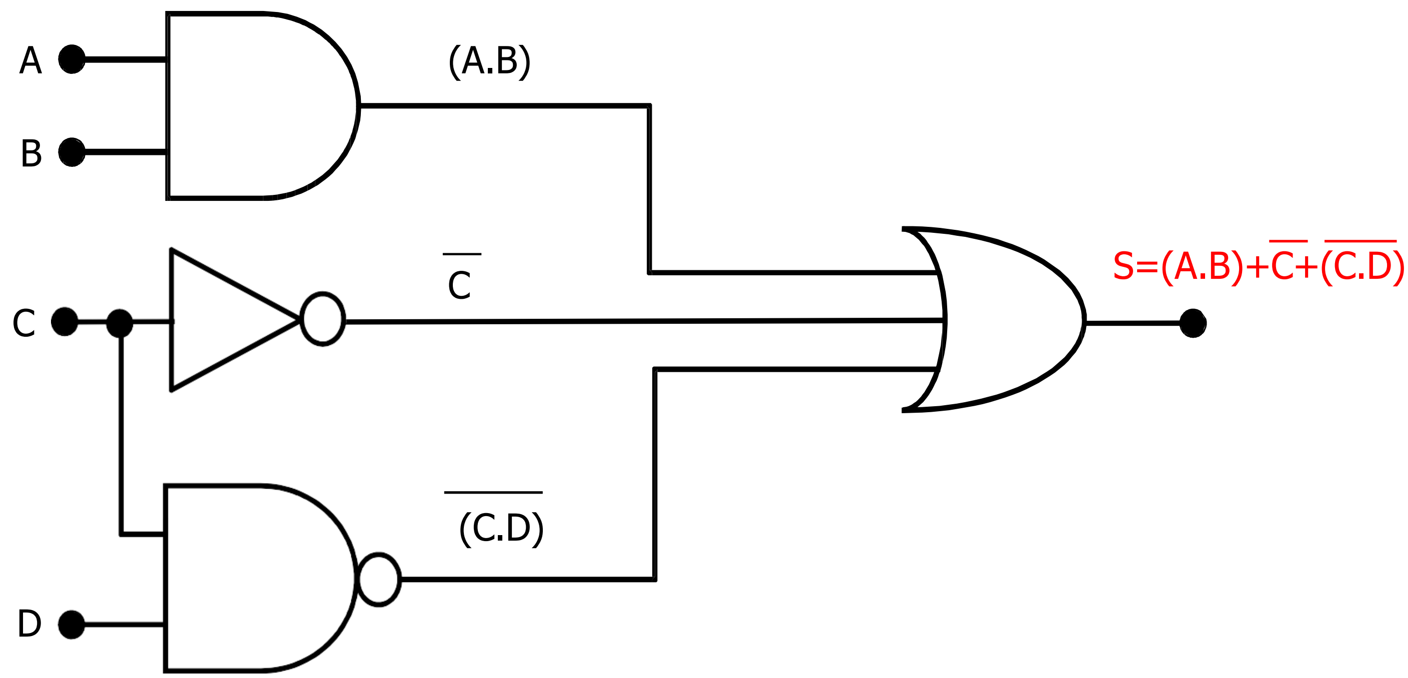
Expressões Booleanas Geradas por Circuitos Lógicos

<u>EXEMPLO 2:</u> Determinar a expressão booleana característica do circuito:



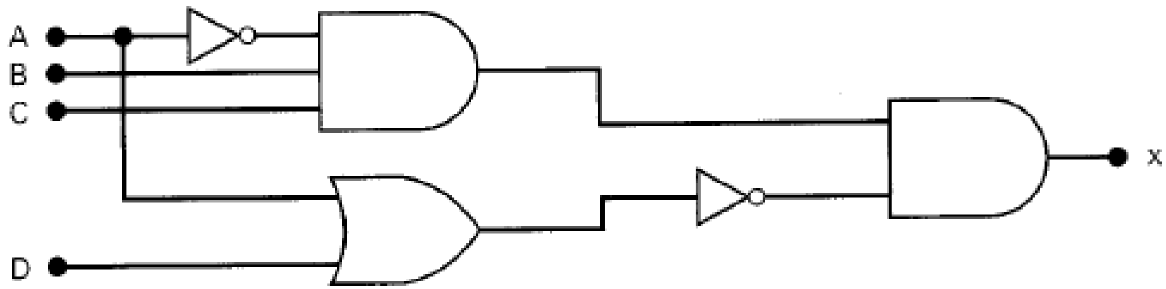
Expressões Booleanas Geradas por Circuitos Lógicos

<u>EXEMPLO 2:</u> Determinar a expressão booleana característica do circuito:

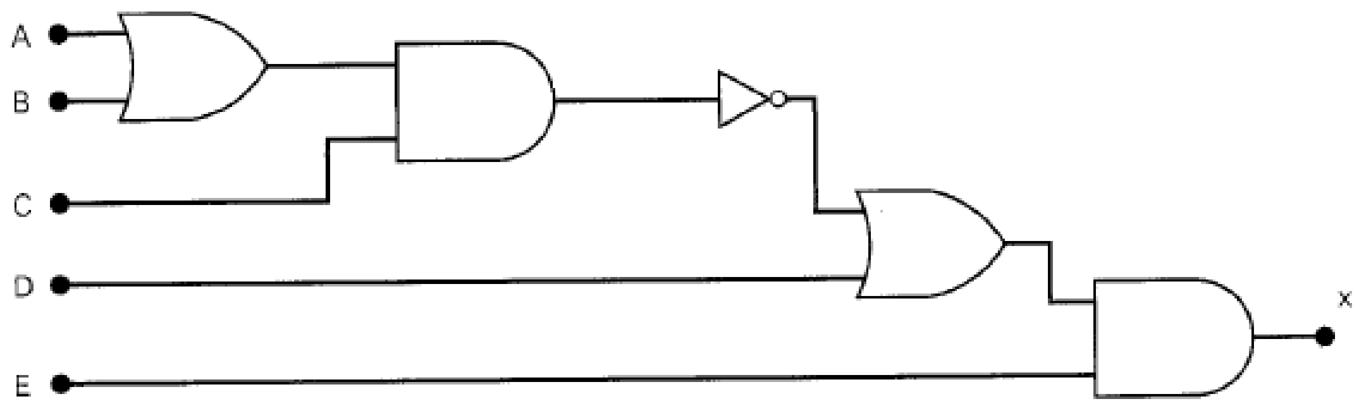


Expressões Booleanas Geradas por Circuitos Lógicos

EXEMPLO 3: Determinar a expressão booleana característica do circuito:



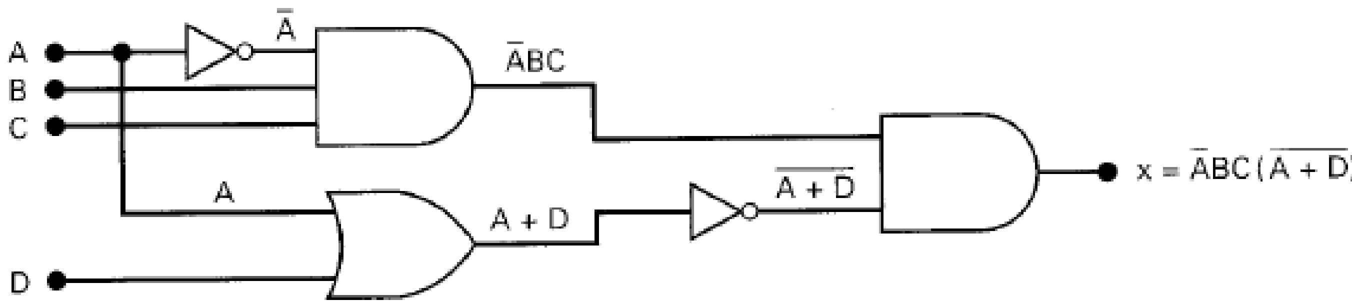
(a)



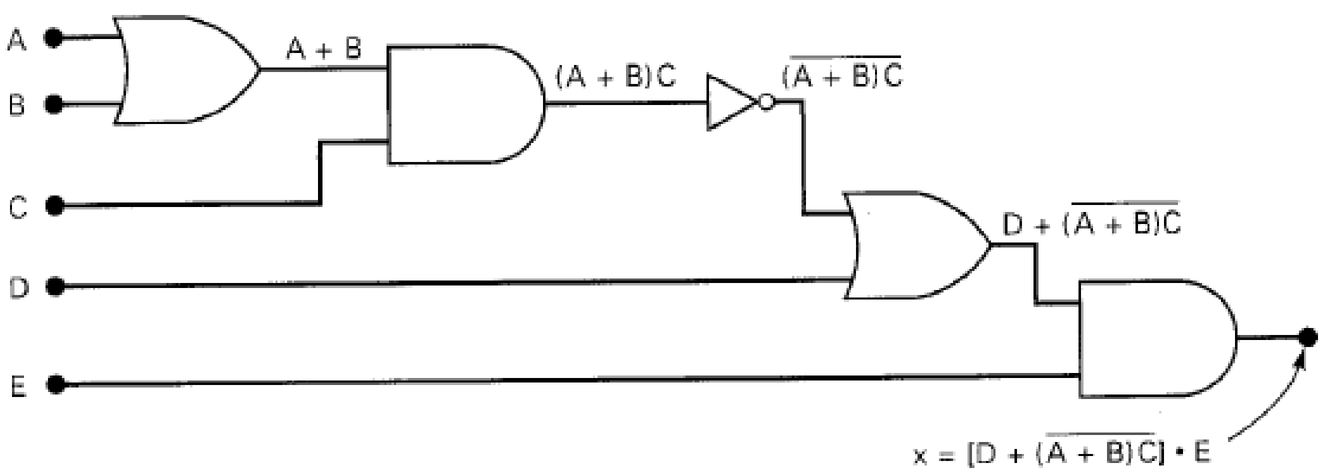
(b)

Expressões Booleanas Geradas por Circuitos Lógicos

EXEMPLO 3: Determinar a expressão booleana característica do circuito:



(a)



(b)

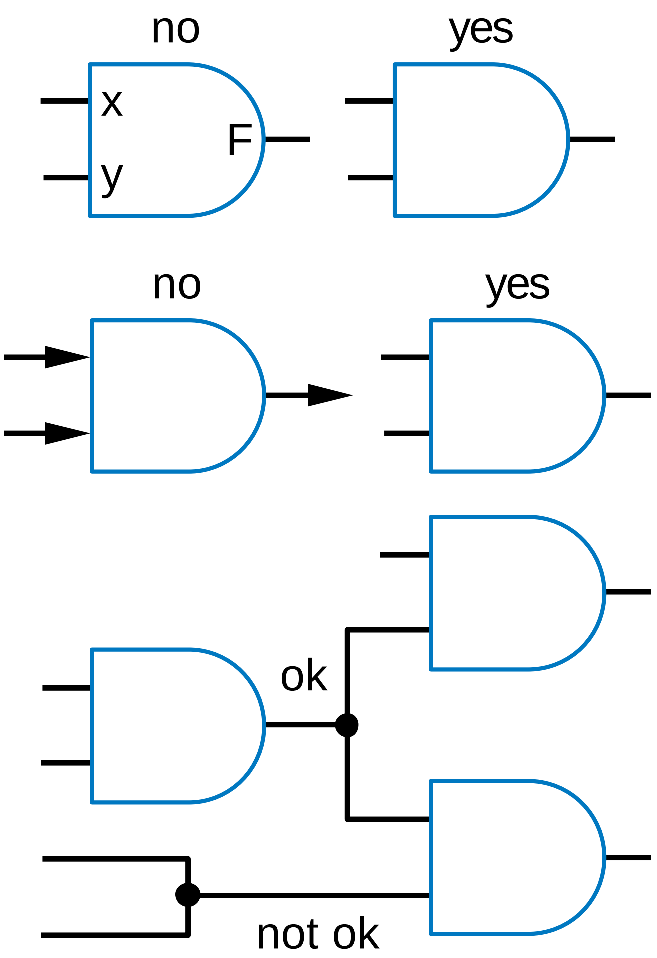
Circuitos Gerados por Expressões Booleanas

- Até o momento, vimos como obter uma expressão característica a partir de um circuito.

- Também é possível **obter um circuito lógico, dada uma expressão booleana.**

- Nesse caso, como na aritmética elementar, parênteses têm maior prioridade, seguidos pela multiplicação (função E) e, por último, pela soma (função OU).

Algumas convenções



Circuitos Gerados por Expressões Booleanas

- **EXEMPLO 1:** Seja a expressão: $S = (A+B).C.(B+D)$.

- Vamos separar as subfórmulas da expressão, ou seja:

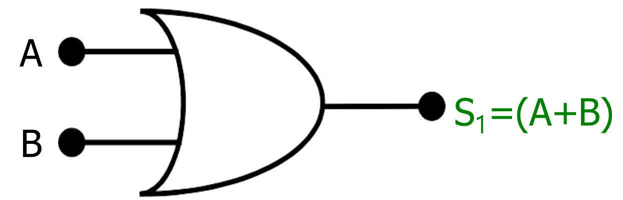
$$S = (A+B) \; . \; C \; . \; (B+D)$$

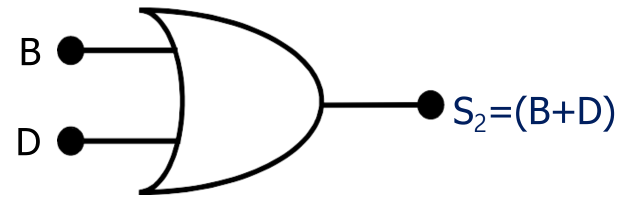
- Dentro do primeiro parêntese temos a soma booleana $S_1 = (A+B)$, portanto o circuito que executa esse parêntese será uma porta OU.

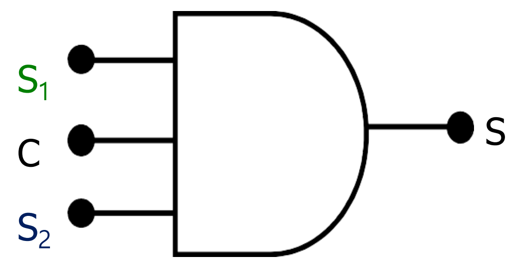
- Dentro do segundo parêntese temos a soma booleana $S_2 = (B+D)$. Novamente, o circuito que executa esse parêntese será uma porta OU.

- Portanto, temos: $S = S_1 \; . \; C \; . \; S_2$

- Agora temos uma multiplicação booleana e o circuito que a executa é uma porta E.

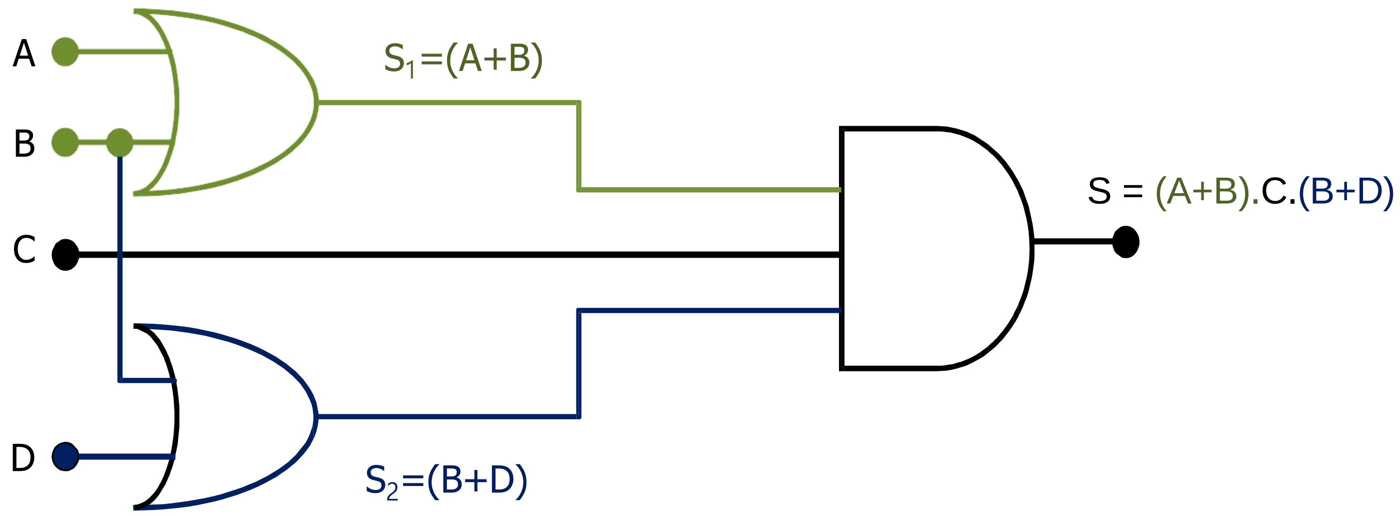






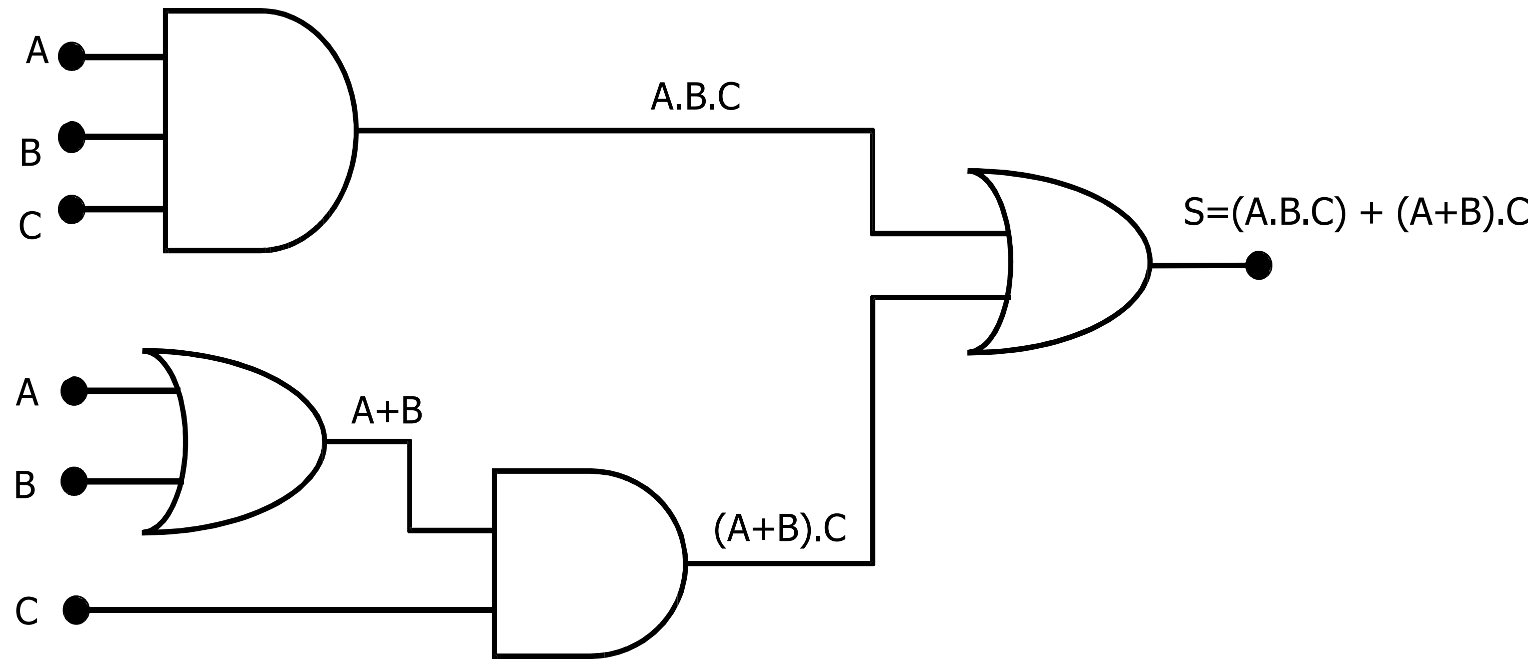
Circuitos Gerados por Expressões Booleanas

Portanto, o circuito completo é:



Circuitos Gerados por Expressões Booleanas

- **<u>EXEMPLO 2:</u>** Desenhe o circuito lógico que executa a seguinte expressão booleana: S = (A.B.C) + (A+B).C



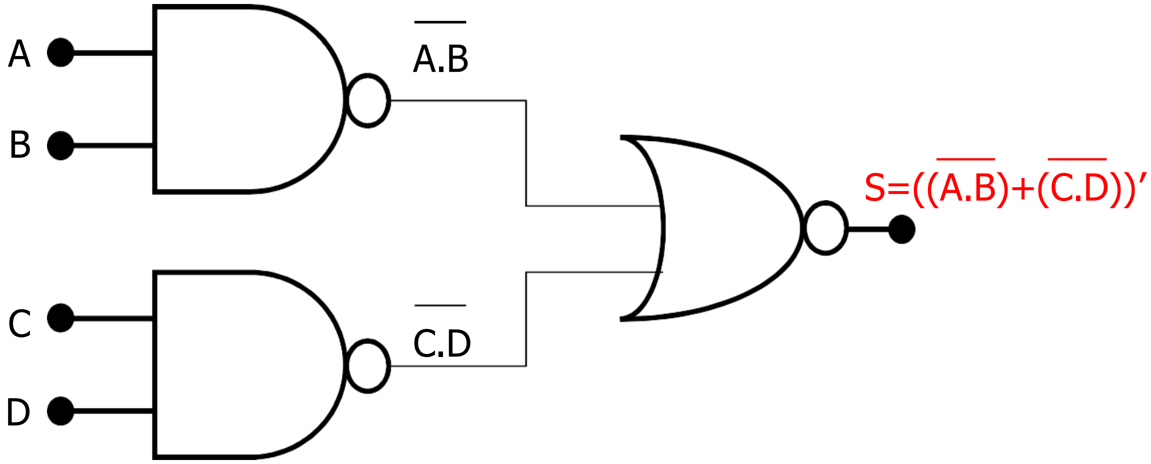
DICAS !

- É importante lembrar que as entradas que representam a mesma variável estão interligadas.
- Contudo o desenho sem interligações facilita a interpretação do circuito.

Circuitos Gerados por Expressões Booleanas

- **EXEMPLO 3:** Desenhe o circuito lógico cuja expressão característica é:

$$S = (\overline{AB} + \overline{CD})'$$



Expressões ou Circuitos representados por Tabelas Verdade

- Uma forma de estudar uma função booleana consiste em utilizar sua tabela verdade.

- Como visto anteriormente, há uma equivalência entre o circuito lógico e sua expressão característica.

✓ **Podemos obter expressões a partir dos circuitos.**
✓ **Podemos obter um circuito a partir de sua expressão.**

- Uma tabela verdade representa o comportamento tanto do circuito como de sua expressão característica.

- A tabela verdade **relaciona todas as combinações possíveis para os níveis lógicos da entrada com o correspondente nível lógico da saída.**

Como obter a Tabela Verdade a partir de uma Expressão

- Colocar todas as possibilidades (interpretações) para as variáveis de entrada.

✓ *Lembrar que para N variáveis de entrada, há 2^N possibilidades.*

- Adicionar colunas para cada subfórmula da expressão.

- Preencher cada coluna com seus resultados.

- Adicionar uma coluna para o resultado final.

- Preencher essa coluna com o resultado final.

Obtendo a Tabela Verdade a partir de uma Expressão

- Como há 4 variáveis de entrada (A, B, C, D), há 2^4 = 16 interpretações.
  - ✓ Primeira Coluna (D) - LSB: Variação 1 zero, 1 um

A	B	C	D
			0
			1
			0
			1
			0
			1
			0
			1
			0
			1
			0
			1
			0
			1
			0
			1

Obtendo a Tabela Verdade a partir de uma Expressão

- Como há 4 variáveis de entrada (A, B, C, D), há $2^4 = 16$ interpretações.

✓ Primeira Coluna (D) - LSB:

Variação 1 zero, 1 um

✓ Segunda Coluna (C):

Variação 2 zeros, 2 uns

A	B	C	D
		0	0
		0	1
		1	0
		1	1
		0	0
		0	1
		1	0
		1	1
		0	0
		0	1
		1	0
		1	1
		0	0
		0	1
		1	0
		1	1

Obtendo a Tabela Verdade a partir de uma Expressão

- Como há 4 variáveis de entrada (A, B, C, D), há $2^4 = 16$ interpretações.

✓ Primeira Coluna (D) - LSB:

Variação 1 zero, 1 um

✓ Segunda Coluna (C):

Variação 2 zeros, 2 uns

✓ Terceira Coluna (B):

Variação 4 zeros, 4 uns

A	B	C	D
	0	0	0
	0	0	1
	0	1	0
	0	1	1
	1	0	0
	1	0	1
	1	1	0
	1	1	1
	0	0	0
	0	0	1
	0	1	0
	0	1	1
	1	0	0
	1	0	1
	1	1	0
	1	1	1

Obtendo a Tabela Verdade a partir de uma Expressão

- Como há 4 variáveis de entrada (A, B, C, D), há $2^4 = 16$ interpretações.

✓ Primeira Coluna (D) - LSB:

Variação 1 zero, 1 um

✓ Segunda Coluna (C):

Variação 2 zeros, 2 uns

✓ Terceira Coluna (B):

Variação 4 zeros, 4 uns

✓ Quarta Coluna (A) - MSB:

Variação 8 zeros, 8 uns

A	B	C	D
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1

Obtendo a Tabela Verdade a partir de uma Expressão

4 variáveis de entrada

2^4 = 16 números
(0 a 15)

	A	B	C	D
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

Obtendo a Tabela Verdade a partir de uma Expressão

EXEMPLO 1: Considere a expressão

S = A.B.C + A.D + A.B.D

A seguir, adicionar uma coluna para cada subfórmula de S, além de uma coluna para o resultado final S.

A.B.C
A.D
A.B.D

A	B	C	D	A.B.C	A.D	A.B.D	S
0	0	0	0				
0	0	0	1				
0	0	1	0				
0	0	1	1				
0	1	0	0				
0	1	0	1				
0	1	1	0				
0	1	1	1				
1	0	0	0				
1	0	0	1				
1	0	1	0				
1	0	1	1				
1	1	0	0				
1	1	0	1				
1	1	1	0				
1	1	1	1				

Obtendo a Tabela Verdade a partir de uma Expressão

S = A.B.C + A.D + A.B.D

A seguir, adicionar uma coluna para cada subfórmula de S, além de uma coluna para o resultado final S.

A.B.C
A.D
A.B.D

Preencher cada coluna com seu respectivo resultado.

A	B	C	D	A.B.C	A.D	A.B.D	S
0	0	0	0				
0	0	0	1				
0	0	1	0				
0	0	1	1				
0	1	0	0				
0	1	0	1				
0	1	1	0				
0	1	1	1				
1	0	0	0				
1	0	0	1				
1	0	1	0				
1	0	1	1				
1	1	0	0				
1	1	0	1				
1	1	1	0	1			
1	1	1	1	1			

Obtendo a Tabela Verdade a partir de uma Expressão

S = A.B.C + A.D + A.B.D

A seguir, adicionar uma coluna para cada subfórmula de S, além de uma coluna para o resultado final S.

A.B.C
A.D
A.B.D

Preencher cada coluna com seu respectivo resultado.

A	B	C	D	A.B.C	A.D	A.B.D	S
0	0	0	0	0			
0	0	0	1	0			
0	0	1	0	0			
0	0	1	1	0			
0	1	0	0	0			
0	1	0	1	0			
0	1	1	0	0			
0	1	1	1	0			
1	0	0	0	0			
1	0	0	1	0			
1	0	1	0	0			
1	0	1	1	0			
1	1	0	0	0			
1	1	0	1	0			
1	1	1	0	1			
1	1	1	1	1			

Obtendo a Tabela Verdade a partir de uma Expressão

S = A.B.C + A.D + A.B.D

A seguir, adicionar uma coluna para cada subfórmula de S, além de uma coluna para o resultado final S.

A.B.C
A.D
A.B.D

Preencher cada coluna com seu respectivo resultado.

A	B	C	D	A.B.C	A.D	A.B.D	S
0	0	0	0	0			
0	0	0	1	0			
0	0	1	0	0			
0	0	1	1	0			
0	1	0	0	0			
0	1	0	1	0			
0	1	1	0	0			
0	1	1	1	0			
1	0	0	0	0			
1	0	0	1	0	1		
1	0	1	0	0			
1	0	1	1	0	1		
1	1	0	0	0			
1	1	0	1	0	1		
1	1	1	0	1			
1	1	1	1	1	1		

Obtendo a Tabela Verdade a partir de uma Expressão

$S = A.B.C + A.D + A.B.D$

A seguir, adicionar uma coluna para cada subfórmula de S, além de uma coluna para o resultado final S.

A.B.C

A.D

A.B.D

Preencher cada coluna com seu respectivo resultado.

A	B	C	D	A.B.C	A.D	A.B.D	S
0	0	0	0	0	0		
0	0	0	1	0	0		
0	0	1	0	0	0		
0	0	1	1	0	0		
0	1	0	0	0	0		
0	1	0	1	0	0		
0	1	1	0	0	0		
0	1	1	1	0	0		
1	0	0	0	0	0		
1	0	0	1	0	1		
1	0	1	0	0	0		
1	0	1	1	0	1		
1	1	0	0	0	0		
1	1	0	1	0	1		
1	1	1	0	1	0		
1	1	1	1	1	1		

Obtendo a Tabela Verdade a partir de uma Expressão

S = A.B.C + A.D + A.B.D

A seguir, adicionar uma coluna para cada subfórmula de S, além de uma coluna para o resultado final S.

A.B.C
A.D
A.B.D

Preencher cada coluna com seu respectivo resultado.

A	B	C	D	A.B.C	A.D	A.B.D	S
0	0	0	0	0	0		
0	0	0	1	0	0		
0	0	1	0	0	0		
0	0	1	1	0	0		
0	1	0	0	0	0		
0	1	0	1	0	0		
0	1	1	0	0	0		
0	1	1	1	0	0		
1	0	0	0	0	0		
1	0	0	1	0	1		
1	0	1	0	0	0		
1	0	1	1	0	1		
1	1	0	0	0	0		
1	1	0	1	0	1	1	
1	1	1	0	1	0		
1	1	1	1	1	1	1	

Obtendo a Tabela Verdade a partir de uma Expressão

$S = A.B.C + A.D + A.B.D$

A seguir, adicionar uma coluna para cada subfórmula de S, além de uma coluna para o resultado final S.

A.B.C

A.D

A.B.D

Preencher cada coluna com seu respectivo resultado.

A	B	C	D	A.B.C	A.D	A.B.D	S
0	0	0	0	0	0	0	
0	0	0	1	0	0	0	
0	0	1	0	0	0	0	
0	0	1	1	0	0	0	
0	1	0	0	0	0	0	
0	1	0	1	0	0	0	
0	1	1	0	0	0	0	
0	1	1	1	0	0	0	
1	0	0	0	0	0	0	
1	0	0	1	0	1	0	
1	0	1	0	0	0	0	
1	0	1	1	0	1	0	
1	1	0	0	0	0	0	
1	1	0	1	0	1	1	
1	1	1	0	1	0	0	
1	1	1	1	1	1	1	

Obtendo a Tabela Verdade a partir de uma Expressão

$S = A.B.C + A.D + A.B.D$

A seguir, adicionar uma coluna para cada subfórmula de S, além de uma coluna para o resultado final S.

A.B.C
A.D
A.B.D

Preencher cada coluna com seu respectivo resultado.

Por último, preencher a coluna do resultado final.

A	B	C	D	A.B.C	A.D	A.B.D	S
0	0	0	0	0	0	0	
0	0	0	1	0	0	0	
0	0	1	0	0	0	0	
0	0	1	1	0	0	0	
0	1	0	0	0	0	0	
0	1	0	1	0	0	0	
0	1	1	0	0	0	0	
0	1	1	1	0	0	0	
1	0	0	0	0	0	0	
1	0	0	1	0	1	0	1
1	0	1	0	0	0	0	
1	0	1	1	0	1	0	1
1	1	0	0	0	0	0	
1	1	0	1	0	1	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	1	1	1

Obtendo a Tabela Verdade a partir de uma Expressão

S = A.B.C + A.D + A.B.D

A seguir, adicionar uma coluna para cada subfórmula de S, além de uma coluna para o resultado final S.

A.B.C

A.D

A.B.D

Preencher cada coluna com seu respectivo resultado.

Por último, preencher a coluna do resultado final.

A	B	C	D	A.B.C	A.D	A.B.D	S
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	1	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	0	0	0	0	0
0	1	0	1	0	0	0	0
0	1	1	0	0	0	0	0
0	1	1	1	0	0	0	0
1	0	0	0	0	0	0	0
1	0	0	1	0	1	0	1
1	0	1	0	0	0	0	0
1	0	1	1	0	1	0	1
1	1	0	0	0	0	0	0
1	1	0	1	0	1	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	1	1	1

Obtendo a Tabela Verdade a partir de uma Expressão

- **EXEMPLO 2:**

Encontre a tabela verdade da expressão

$$S = \bar{A} + B + A.B.C'$$

Obtendo a Tabela Verdade a partir de uma Expressão

- **EXEMPLO 2:**

Encontre a tabela verdade da expressão

$$S = \bar{A} + B + A.B.C'$$

A	B	C	Ā	C’	A.B.C’	S
0	0	0	1	1		
0	0	1	1	0		
0	1	0	1	1		
0	1	1	1	0		
1	0	0	0	1		
1	0	1	0	0		
1	1	0	0	1		
1	1	1	0	0		

Obtendo a Tabela Verdade a partir de uma Expressão

- **EXEMPLO 2:**

Encontre a tabela verdade da expressão

$$S = \bar{A} + B + A.B.C'$$

A	B	C	Ā	C’	A.B.C’	S
0	0	0	1	1	0	
0	0	1	1	0	0	
0	1	0	1	1	0	
0	1	1	1	0	0	
1	0	0	0	1	0	
1	0	1	0	0	0	
1	1	0	0	1	1	
1	1	1	0	0	0	

Obtendo a Tabela Verdade a partir de uma Expressão

- **EXEMPLO 2:**

Encontre a tabela verdade da expressão

$$S = \bar{A} + B + A.B.C'$$

A	B	C	Ā	C'	A.B.C'	S
0	0	0	1	1	0	1
0	0	1	1	0	0	1
0	1	0	1	1	0	1
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	1	0	0	0	0
1	1	0	0	1	1	1
1	1	1	0	0	0	1

Obtendo a Tabela Verdade a partir de uma Expressão

<u>EXEMPLO 3:</u> Montar a tabela verdade da expressão:

$$S = A.B.C + A.B'.C + A'.B'.C + A'.B'.C'$$

Obtendo a Tabela Verdade a partir de uma Expressão

<u>EXEMPLO 3:</u> Montar a tabela verdade da expressão:

$$S = A.B.C + A.B'.C + A'.B'.C + A'.B'.C'$$

A	B	C	A'	B'	C'	A.B.C	A.B'.C	A'.B'.C	A'.B'.C'	S
0	0	0	1	1	1					
0	0	1	1	1	0					
0	1	0	1	0	1					
0	1	1	1	0	0					
1	0	0	0	1	1					
1	0	1	0	1	0					
1	1	0	0	0	1					
1	1	1	0	0	0					

Obtendo a Tabela Verdade a partir de uma Expressão

EXEMPLO 3: Montar a tabela verdade da expressão:

$$S = A.B.C + A.B'.C + A'.B'.C + A'.B'.C'$$

A	B	C	A'	B'	C'	A.B.C	A.B'.C	A'.B'.C	A'.B'.C'	S
0	0	0	1	1	1	0	0	0	1	1
0	0	1	1	1	0	0	0	1	0	1
0	1	0	1	0	1	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0	0
1	0	0	0	1	1	0	0	0	0	0
1	0	1	0	1	0	0	1	0	0	1
1	1	0	0	0	1	0	0	0	0	0
1	1	1	0	0	0	1	0	0	0	1

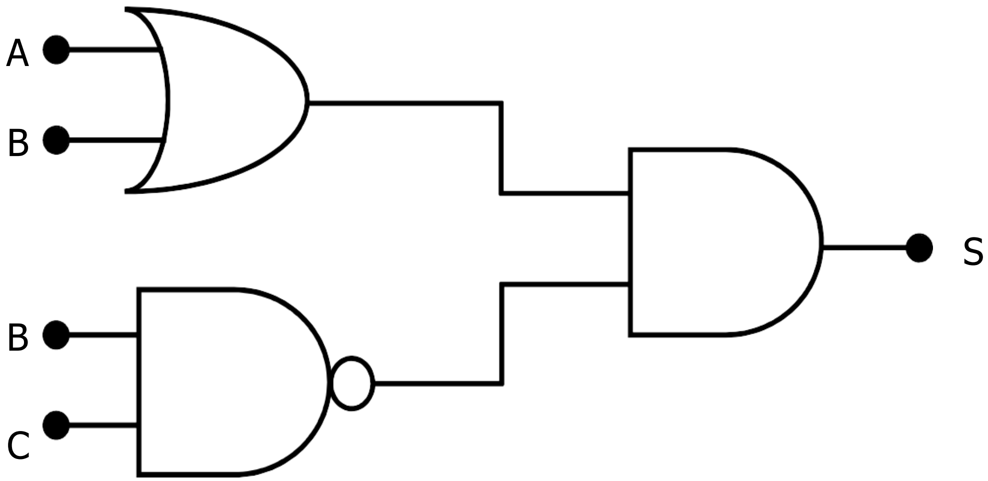
Obtendo a Tabela Verdade a partir de um Circuito

- De forma análoga, é possível estudar o comportamento de um circuito por meio da sua tabela verdade.

- **Dado um circuito**, é necessário **extrair sua expressão** característica; a partir da sua expresão booleana é possível **montar a tabela verdade** correspondente.

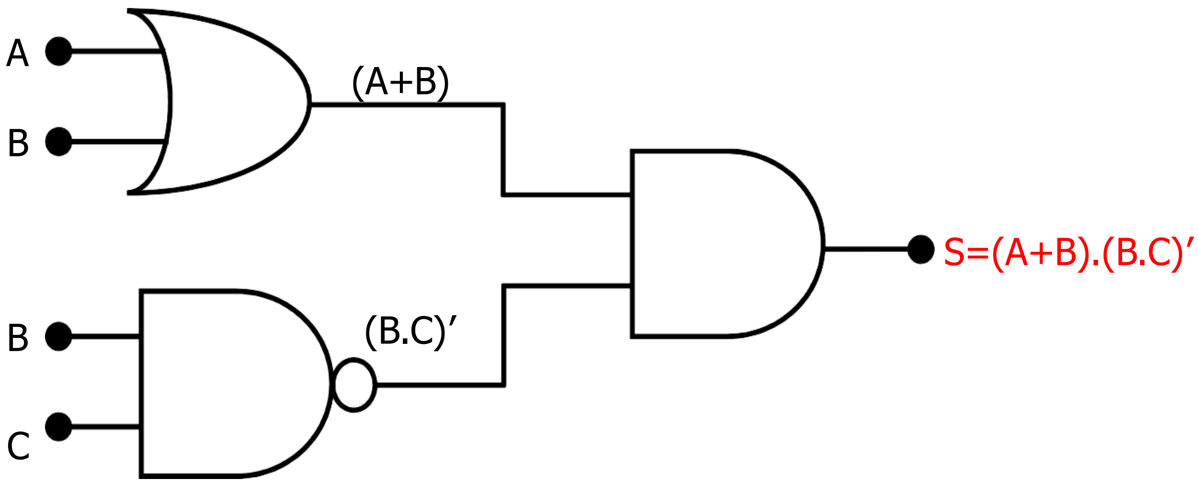
Obtendo a Tabela Verdade a partir de um Circuito

A partir do circuito, construir sua tabela verdade:



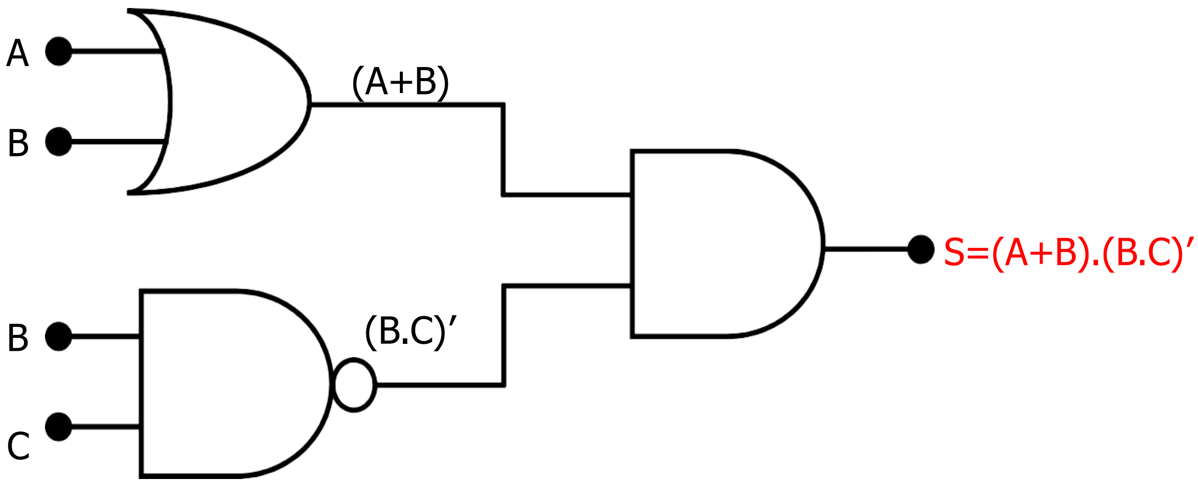
Obtendo a Tabela Verdade a partir de um Circuito

A partir do circuito, construir sua tabela verdade:



Obtendo a Tabela Verdade a partir de um Circuito

A partir do circuito, construir sua tabela verdade:



Extraímos sua expressão característica

S = (A+B) . (B.C)’

Obtendo a Tabela Verdade a partir de um Circuito

- A partir da expressão:

  $S = (A+B) . (B.C)'$

- Obtém-se a tabela verdade, como anteriormente explicado.

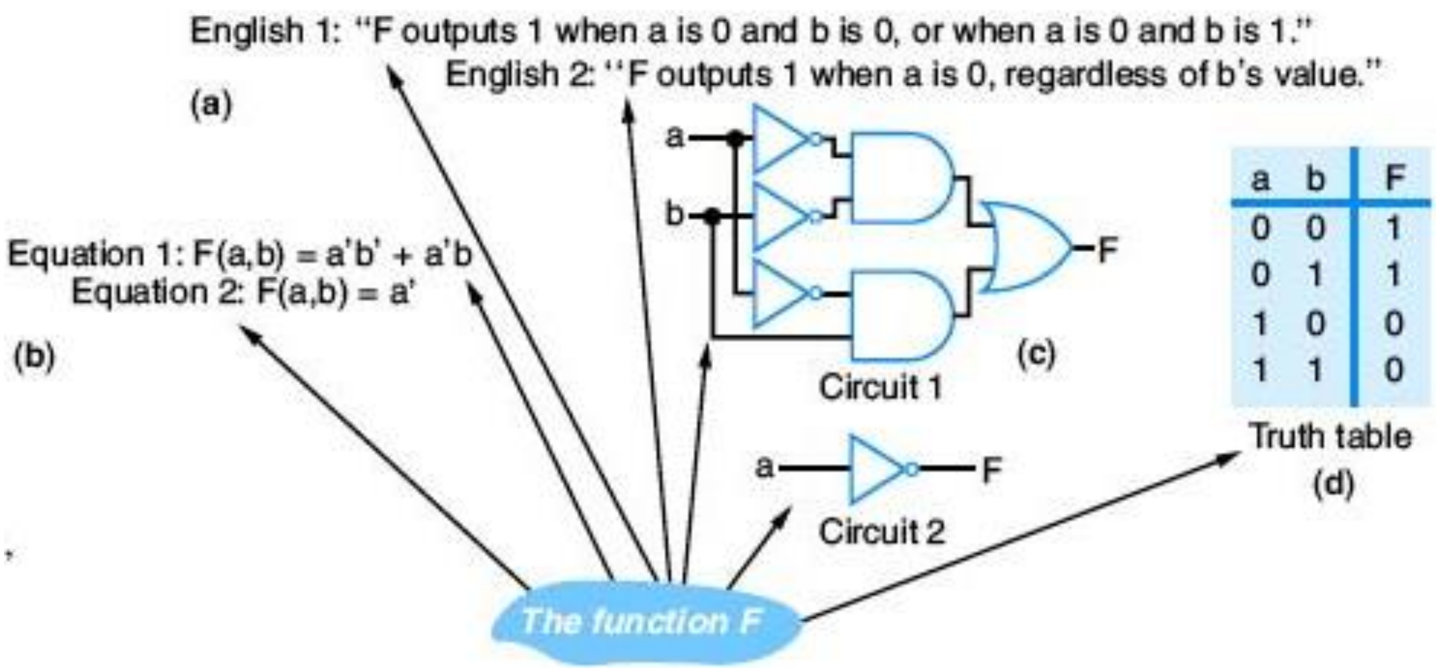
- Ou podemos atribuir diretamente valores de 000 a 111 para as entradas A, B, C e, a partir do circuito, gerar a saída registrando-a na tabela verdade.

A	B	C	A+B	B.C	(B.C)’	S
0	0	0	0	0	1	0
0	0	1	0	0	1	0
0	1	0	1	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	0	1	1
1	1	0	1	0	1	1
1	1	1	1	1	0	0

Formas Canônicas

Representações de Funções Booleanas

- **Função booleana:**
  - Mapeamento de cada uma das combinações possíveis de valores da função (as entradas) para 0 ou 1 (a saída).



Representações de Funções Booleanas

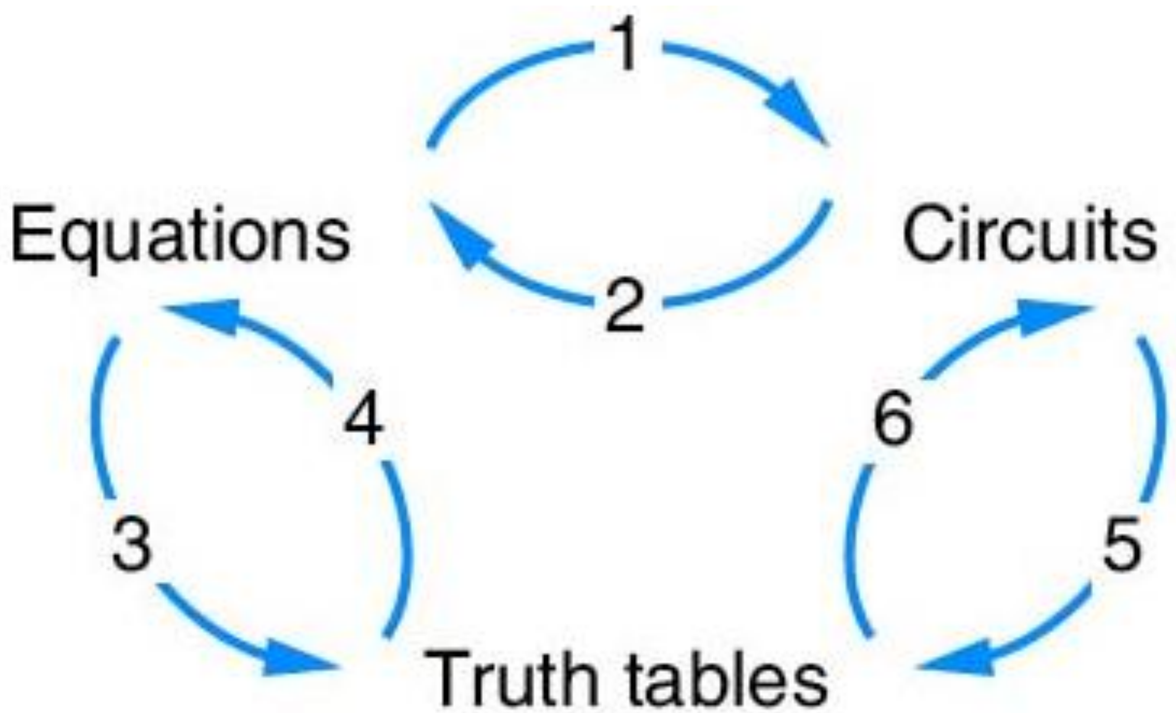
- **Equações**
  - Enunciados matemáticos que igualam uma expressão à outra (descreve uma função).
  - Exemplo: $F(a,b) = a'b' + a'b \rightarrow F(a,b) = a'$

- **Circuitos**
  - Mapeamento predefinido entre valores de entrada e de saída.
  - Vantagens:
    - Implementação física real e facilidade no entendimento de uma função.

- **Tabela-verdade**
  - Apenas 1 representação por função!!!

Inputs		Output
a	b	F
0	0	
0	1	
1	0	
1	1	

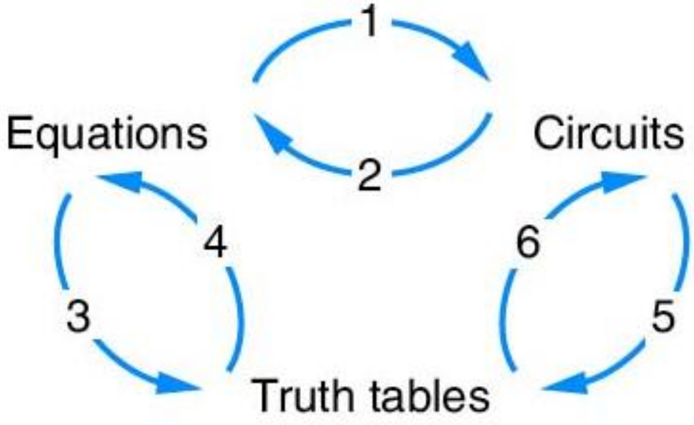
Convertendo entre Representações

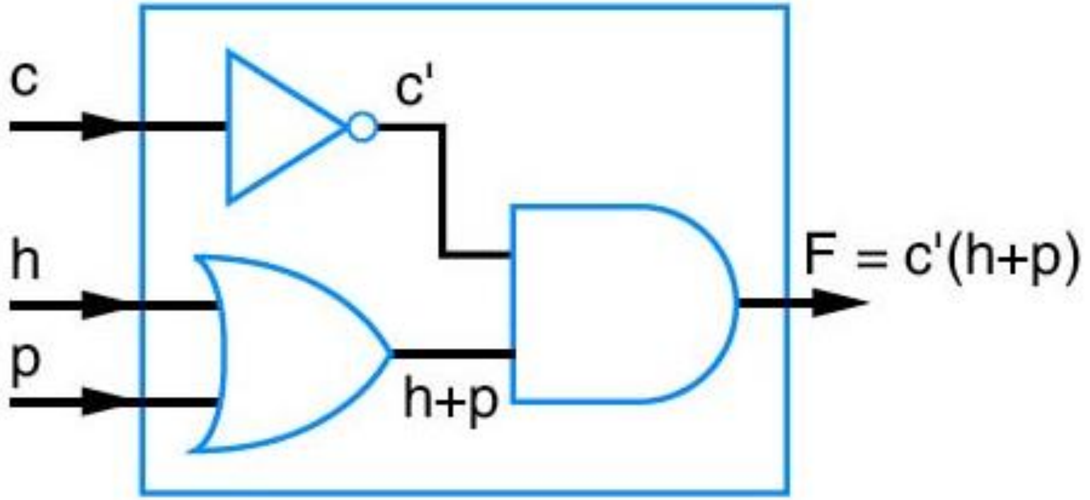


Equações → Circuitos → Tabela Verdade

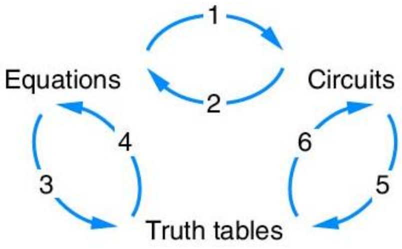
1. **Equações em Circuitos:**
   – Conforme visto anteriormente
   – Uma porta para cada operador

2. **Circuitos em Equações:**





Equações → Circuitos → Tabela Verdade



3. **Equações em Tabelas-Verdade:**
   – Preparar uma tabela-verdade de acordo com o número de entradas.
   – Atribuir os níveis lógicos das respectivas saídas.

4. **Tabelas-Verdade em Equações:**
   – Basta criar um termo de produto para cada 1 (ou um termo de soma para cada 0) da saída e aplicar o operador OR (ou o operador AND) a estes termos.

   – No exemplo:
     - F = a’b’ + a’b

Inputs		Outputs	Term
a	b	F	F = sum of
0	0	1	a' b'
0	1	1	a' b
1	0	0	
1	1	0	

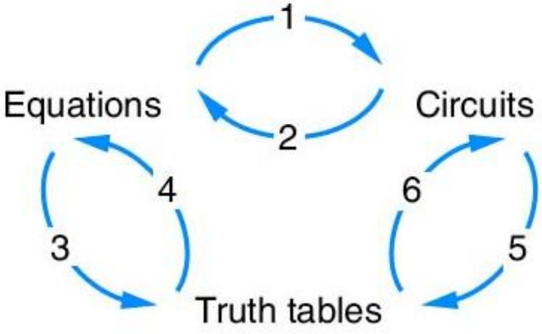
Equações → Circuitos → Tabela Verdade

5. Circuitos em Tabela-Verdade

– Circuito → Equação → Tabela-Verdade

6. Tabela-Verdade em Circuitos

– Tabela-Verdade→ Equação → Circuito



Representação Padrão

- Representação em tabela-verdade é única para uma função booleana → **Representação Padrão.**

- A representação padrão é uma forma de verificar se duas funções são equivalentes.

- E quando uma função tem muitas entradas?
  – Por exemplo, se temos 32 entradas, teremos 2^{32} (cerca de 4 bilhões) linhas na tabela!!!

Representação Padrão

EXISTE UMA REPRESENTAÇÃO MAIS COMPACTA DE UMA FUNÇÃO BOOLEANA?

Formas Canônicas

- Representação padrão que descreva apenas as situações nas quais a saída é 1 ou 0.

Expressões booleanas são normalmente representadas em um formato padrão, conhecido como forma canônica.

Existem dois tipos equivalentes de formas canônicas:

Soma de produtos (SOP)

Produto de somas (POS)

Forma Canônica – Soma de Produtos (SOP)

- Também conhecida como forma normal disjuntiva ou expansão de mintermos.

- **Mintermo:** um termo de produto que contém todas as literais da função exatamente uma vez e tem o valor 1 para cada atribuição das variáveis.

- Dada uma tabela verdade, cada linha onde a função de saída possui valor igual a 1 contribui com um termo.

A	B	C	F	
0	0	0	0	
0	0	1	1	$\bar{A}\bar{B}C$
0	1	0	0	
0	1	1	1	$\bar{A}BC$
1	0	0	0	
1	0	1	1	$A\bar{B}C$
1	1	0	1	$AB\bar{C}$
1	1	1	1	ABC

- Podemos interpretar as variáveis A, B, C como uma palavra de três bits, onde:
  - A é o bit mais significativo.
  - C é o bit menos significativo.

Literais: variáveis complementadas (invertidas) ou não-complementadas (não-invertidas)

Forma Canônica – Soma de Produtos (SOP)

- Cada termo é formado por operações AND entre:
  - as variáveis de entrada cujo valor é 1, e
  - os complementos das variáveis de entrada cujo valor é 0.

Variáveis não complementadas são associadas com 1 e complementadas com 0.

- Os termos são concatenados por operações OR, a fim de compor a expressão canônica.

A	B	C	F	
0	0	0	0	
0	0	1	1	$\bar{A}\bar{B}C$
0	1	0	0	
0	1	1	1	$\bar{A}BC$
1	0	0	0	
1	0	1	1	$A\bar{B}C$
1	1	0	1	$AB\bar{C}$
1	1	1	1	ABC

Forma canônica SOP:

$$f(A,B,C) = \bar{A}\bar{B}C + \bar{A}BC + A\bar{B}C + AB\bar{C} + ABC$$

SOP: soma de produtos em que todos os produtos são mintermos e nenhum mintermo é repetido !!

Forma Canônica – Soma de Produtos (SOP)

- Assim, a expressão canônica da SOP pode ser reescrita de forma mais compacta:

$$f(A,B,C) = \bar{A}\bar{B}C + \bar{A}BC + A\bar{B}C + AB\bar{C} + ABC$$

$$f(A,B,C) = \sum m(001,011,101,110,111)$$

A	B	C	F	
0	0	0	0	
0	0	1	1	$\bar{A}\bar{B}C$
0	1	0	0	
0	1	1	1	$\bar{A}BC$
1	0	0	0	
1	0	1	1	$A\bar{B}C$
1	1	0	1	$AB\bar{C}$
1	1	1	1	ABC

$$\boxed{f(A,B,C) = \sum m(1,3,5,6,7)}$$

Termo Produto	mintermo
$\bar{X}\cdot\bar{Y}$	m_0
$\bar{X}\cdot Y$	m_1
$X\cdot\bar{Y}$	m_2
$X\cdot Y$	m_3

Mintermos para duas variáveis

Termo Produto	mintermo
$\bar{X}\cdot\bar{Y}\cdot\bar{Z}$	m_0
$\bar{X}\cdot\bar{Y}\cdot Z$	m_1
$\bar{X}\cdot Y\cdot\bar{Z}$	m_2
$\bar{X}\cdot Y\cdot Z$	m_3
$X\cdot\bar{Y}\cdot\bar{Z}$	m_4
$X\cdot\bar{Y}\cdot Z$	m_5
$X\cdot Y\bar{Z}$	m_6
$X\cdot Y\cdot Z$	m_7

Mintermos para três variáveis

Forma Canônica – Produto de Somas (POS)

- Também conhecida como forma normal conjuntiva ou expansão de maxtermos.

- **Maxtermo:** um termo de soma que contém todas as literais da função exatamente uma vez e tem o valor 0 para cada atribuição das variáveis.

- Dada uma tabela verdade, cada linha onde a função de saída possui valor igual a zero contribui com um termo (maxtermo).

A	B	C	F	
0	0	0	0	$A + B + C$
0	0	1	1	
0	1	0	0	$A + \bar{B} + C$
0	1	1	1	
1	0	0	0	$\bar{A} + B + C$
1	0	1	1	
1	1	0	1	
1	1	1	1	

- Podemos interpretar as variáveis A, B, C como uma palavra de três bits, onde:
  - A é o bit mais significativo.
  - C é o bit menos significativo.

Pode ser uma representação mais compacta se a função apresenta um número pequeno de zeros.

Forma Canônica – Produto de Somas (POS)

- Cada termo é formado por operações OR entre:
  - as variáveis de entrada cujo valor é 0, e
  - os complementos das variáveis de entrada cujo valor é 1.

Variáveis não complementadas são associadas com 0 e complementadas com 1 (inverso dos Mintermos).

- Os termos são concatenados por operações AND, a fim de compor a expressão canônica.

A	B	C	F	
0	0	0	0	$A+B+C$
0	0	1	1	
0	1	0	0	$A+\bar{B}+C$
0	1	1	1	
1	0	0	0	$\bar{A}+B+C$
1	0	1	1	
1	1	0	1	
1	1	1	1	

Forma canônica POS:

$$f(A,B,C) = (A+B+C)\cdot(A+\bar{B}+C)\cdot(\bar{A}+B+C)$$

POS: produtos de somas em cada soma é um maxtermo e nenhum termo é repetido !!

Forma Canônica – Produto de Somas (POS)

Termo Produto	maxtermo
$X+Y+Z$	M_0
$X+Y+\overline{Z}$	M_1
$X+\overline{Y}+\overline{Z}$	M_2
$X+\overline{Y}+Z$	M_3
$\overline{X}+Y+Z$	M_4
$\overline{X}+Y+\overline{Z}$	M_5
$\overline{X}+\overline{Y}+Z$	M_6
$\overline{X}+\overline{Y}+\overline{Z}$	M_7

Maxtermos para três variáveis

- Da mesma forma, a expressão canônica do POS pode ser reescrita de forma mais compacta:

$$f(A,B,C) = (A+B+C)\cdot(A+\bar{B}+C)\cdot(\bar{A}+B+C)$$

$$f(A,B,C) = \prod M(000,010,100)$$

$$\boxed{f(A,B,C) = \prod M(0,2,4)}$$

A	B	C	F	
0	0	0	0	$A+B+C$
0	0	1	1	
0	1	0	0	$A+\bar{B}+C$
0	1	1	1	
1	0	0	0	$\bar{A}+B+C$
1	0	1	1	
1	1	0	1	
1	1	1	1	

Equivalência entre as Formas POS e SOP

- As formas canônicas POS e SOP são equivalentes, pois derivam de uma mesma tabela verdade.

- Assim, dos exemplos anteriores, temos:

$$\boxed{\sum m(1,3,5,6,7) \equiv \prod M(0,2,4)}$$

Um Pouco de Formalismo Matemático

- **<u>Soma de Mintermos:</u>**

$$E(x) = \Sigma\ m\ (\{j \mid f(j) = 1\})$$

m = mintermos

- **<u>Produto de Maxtermos:</u>**

$$E(x) = \Pi\ M\ (\{j \mid f(j) = 0\})$$

M = maxtermos

- Finalmente,

$$\Sigma\ m\ (\{j \mid f(j) = 1\}) = \Pi\ M\ (\{j \mid f(j) = 0\})$$

Organização e Forma Compacta

j	$x_2x_1x_0$	$F(x_2,x_1,x_0)$	Mintermos	Maxtermos
0	000	0	$x_2'x_1'x_0'$	$x_2+x_1+x_0$
1	001	1	$x_2'x_1'x_0$	$x_2+x_1+x_0'$
2	010	1	$x_2'x_1x_0'$	$x_2+x_1'+x_0$
3	011	0	$x_2'x_1x_0$	$x_2+x_1'+x_0'$
4	100	1	$x_2x_1'x_0'$	$x_2'+x_1+x_0$
5	101	1	$x_2x_1'x_0$	$x_2'+x_1+x_0'$
6	110	1	$x_2x_1x_0'$	$x_2'+x_1'+x_0$
7	111	0	$x_2x_1x_0$	$x_2'+x_1'+x_0'$

$$F\,(x_2,x_1,x_0) = \Sigma\, m\,(1,2,4,5,6) = \Pi\, M\,(0,3,7)$$

Como fazer a Conversão SOP $\rightleftarrows$ POS?

- A **soma de mintermos corresponde ao conjunto-um** da função, enquanto que o **produto de maxtermos corresponde ao conjunto-zero.**

- Para converter de uma forma canônica para a outra, **basta incluir os índices que não aparecem** !

- Voltando ao exemplo anterior:

$$F(x_2,x_1,x_0) = \Sigma\, m\,(\mathbf{1,2,4,5,6}) = \Pi\, M\,(\mathbf{0,3,7})$$

Especificações Incompletas (Don't Care)

- Há situações em que a saída produzida por um conjunto particular de entradas não é especificada:
  - A saída 0 ou 1 não afetam o resultado do circuito.
- Tal ocorrência é conhecida como don't care.
- Soma de produtos:

$$f = d(\ldots)$$

- Produto de somas:

$$f = D(\ldots)$$

Exemplo 1:
Especificações Incompletas (Don’t Care)

- Qual a tabela verdade equivalente a

$$f(A,B,C,D) = \sum m(0,2,6,9,13,14) + \sum d(3,8,10)$$

A	B	C	D	f	#
0	0	0	0	1	**0**
0	0	0	1	0	**1**
0	0	1	0	1	**2**
0	0	1	1	X	**3**
0	1	0	0	0	**4**
0	1	0	1	0	**5**
0	1	1	0	1	**6**
0	1	1	1	0	**7**
1	0	0	0	X	**8**
1	0	0	1	1	**9**
1	0	1	0	X	**10**
1	0	1	1	0	**11**
1	1	0	0	0	**12**
1	1	0	1	1	**13**
1	1	1	0	1	**14**
1	1	1	1	0	**15**

Circuitos Combinacionais de Múltiplas Saídas

- Como lidar com um circuito com mais de uma saída?

Basta tratar separadamente cada saída!

Exemplo 2:
Especificações Incompletas (Don’t Care)

- Derive as expressões SOP das saídas de um circuito que recebe como entrada um número binário de 4 bits, entre 0 e 9, e o converte para BCD Excesso de 3.

Dígito	Excesso de 3
0	0011
1	0100
2	0101
3	0110
4	0111
5	1000
6	1001
7	1010
8	1011
9	1100

Exemplo 2:
Especificações Incompletas (Don't Care)

Dígito	Excesso de 3
0	0011
1	0100
2	0101
3	0110
4	0111
5	1000
6	1001
7	1010
8	1011
9	1100

A	B	C	D	S_3	S_2	S_1	S_0
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0
1	0	1	0	X	X	X	X
1	0	1	1	X	X	X	X
1	1	0	0	X	X	X	X
1	1	0	1	X	X	X	X
1	1	1	0	X	X	X	X
1	1	1	1	X	X	X	X

Exemplo 2:
Especificações Incompletas (Don’t Care)

A	B	C	D	S_3
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

Expressão SOP – Saída S_3

$$F = \sum m(5,6,7,9) + \sum d(10,11,12,13,14,15)$$

Exemplo 2:
Especificações Incompletas (Don't Care)

A	B	C	D	S_2
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

Expressão SOP – Saída S_2

SOP:

$$F = \sum m(1,2,3,4,9) + \sum d(10,11,12,13,14,15)$$

Exemplo 2:
Especificações Incompletas (Don't Care)

A	B	C	D	S_1
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

Expressão SOP – Saída S_1

SOP:

$$F = \sum m(0,3,4,7,8) + \sum d(10,11,12,13,14,15)$$

Exemplo 2:
Especificações Incompletas (Don’t Care)

A	B	C	D	S_0
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	0
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

Expressão SOP – Saída S_0

SOP:

$$F = \sum m(0,2,4,6,8) + \sum d(10,11,12,13,14,15)$$

EXERCÍCIOS

Exercício 1: Desenhe a tabela da verdade para a função:

$$E\ (x_2,x_1,x_0) = \Sigma\ m\ (1,3,7) = \Pi\ M\ (0,2,4,5,6)$$

Exercício 2: Descreva as funções abaixo na forma canônica soma de produtos (mintermos):

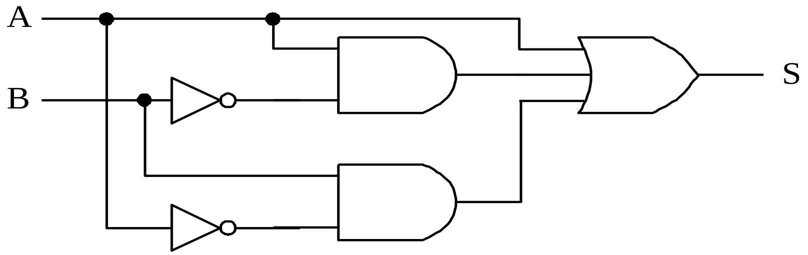
a) F = a´bc + abc´+ ab´c´
b) F = a´bcd + abcd´+ ab´cd´

Exercício 3: Repita o exercício anterior para produto de somas (MAXTERMOS)

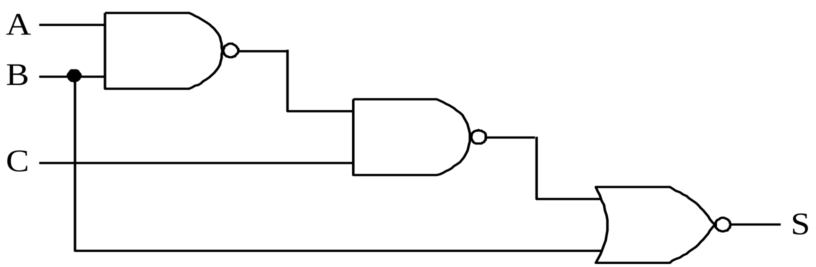
EXERCÍCIOS

Exercício 4: Descreva as saídas dos circuitos abaixo em termos das tabelas da verdade e escreva as expressões equivalentes aos mintermos e maxtermos.

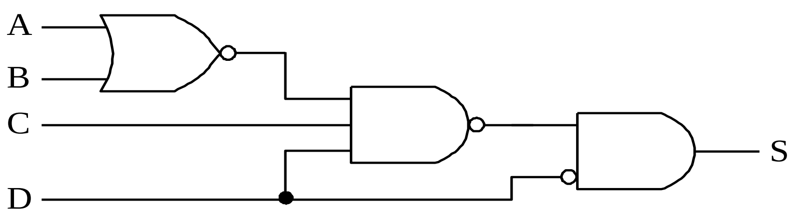
a)



b)

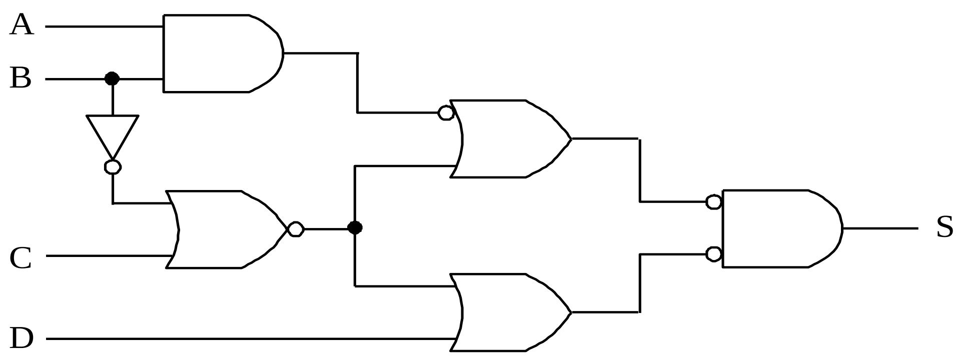


c)

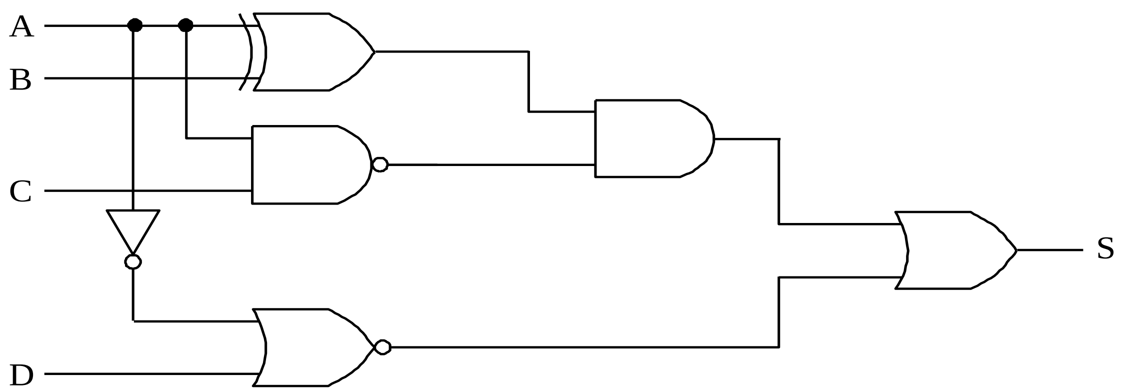


Exercícios

d)



e)



Exercícios

Exercício 5: Descreva as expressões das funções abaixo em termos dos mintermos correspondentes.

a) $S = \overline{\overline{A + B}\,.\,C}$

b) $S = \overline{\overline{A\,.\,B} + C}$

c) $S = \left(\overline{A\,.\,B + C}\right).\overline{D}$

d) $S = \left(\overline{A} + \overline{B}\right).\left(\overline{\overline{A} + \overline{B}}\right)$

e) $S = \left[\left(A \oplus B\right) + \overline{A}\right].\overline{C}$

f) $S = \left[\overline{B} + \left(A \oplus B\right)\right].\overline{A} + \overline{C}$

EXERCÍCIOS

Exercício 6: Descreva as expressões das funções abaixo em termos dos mintermos e maxtermos correspondentes

a)

A	B	C	S
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

b)

A	B	C	S
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

c)

A	B	C	S
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

d)

A	B	C	S
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

e)

A	B	C	D	S
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	0

Dúvidas ??

OBRIGADO PELA ATENÇÃO

Prof. Victor M. Miranda